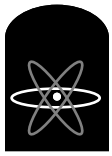
UCRL-ID-112900

The Programmable Logic Controller and Its Application in Nuclear Reactor Systems

J. Palomar
R. Wyman

June 30, 1993



FESSP
Fission Energy and Systems Safety Program

Lawrence Livermore National Laboratory

Work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

The Programmable Logic Controller and Its Application in Nuclear Reactor Systems

J. Palomar

R. Wyman

Manuscript Date: June 30, 1993

Abstract

This document provides recommendations to guide reviewers in the application of Programmable Logic Controllers (PLCs) to the control, monitoring and protection of nuclear reactors. The first topics addressed are system-level design issues, specifically including safety. The document then discusses concerns about the PLC manufacturing organization and the protection system engineering organization. Supplementing this document are two appendices. Appendix A summarizes PLC characteristics. Specifically addressed are those characteristics that make the PLC more suitable for emergency shutdown systems than other electrical/electronic-based systems, as well as characteristics that improve reliability of a system. Also covered are PLC characteristics that may create an unsafe operating environment. Appendix B provides an overview of the use of programmable logic controllers in emergency shutdown systems. The intent is to familiarize the reader with the design, development, test, and maintenance phases of applying a PLC to an ESD system. Each phase is described in detail and information pertinent to the application of a PLC is pointed out.

Contents

Contents—Appendix A: Programmable Logic Controller Characteristics and Safety Considerations

Figures—Appendix A

Contents—Appendix B: Application of Programmable Logic Controllers in Safety Shutdown Systems

Figures—Appendix B

The Programmable Logic Controller and Its Application in Nuclear Reactor Systems

1. INTRODUCTION

1.1. Purpose

The purpose of this document is to outline recommendations for guidance for the review of application of Programmable Logic Controllers (PLCs) to the control, monitoring and protection of nuclear reactors. Several characteristics set PLCs apart from other programmable electronic systems (PESs) or computer systems: they use a deterministic operating system, their primary interfaces are digital and analog sensors and actuating devices, and they typically allow programming to be done with ladder logic diagrams. These diagrams allow the design of the logic of the system to proceed using the traditional ladder logic method of expressing the logic of the system. However, as these systems have evolved, they have begun to include other programming methods. PLC systems are beginning to resemble other general-purpose industrial digital control systems as the distinctions between them have begun to blur.

Because of this trend toward making PLCs more flexible, the methods employed to design and install these systems are tending toward those used in other programmable systems. Thus, the guidance recommended below will resemble the guidance which should be used for more general-purpose programmable systems.

1.2. Scope

In order to make programmable systems more flexible and easier to use for those who are not proficient with programming languages, industry has developed configurable systems such as PLCs. Configurable systems provide a unique approach to designing a PES in that a user of the system selects hardware components and programs the system using a restricted, very-high-level language (referred to as "configuration") to build a system. Configurable systems make implementation of a nuclear power plant protection system easier than it would be if a non-configurable microprocessor-based system was used. This document addresses PLC-based protection systems only.

For a PLC-based protection system two organizations will produce the software and design the hardware—the protection system engineering organization and the PLC manufacturing organization. Guidance concerns are addressed in this document for both organizations. The protection system organization selects the PLC and establishes the protection system requirements, while the PLC manufacturer's organization is concerned with providing a system that appeals to its customers. Since most of the PLC orders come from industry (i.e., auto, food processing, petrochemical), the PLC manufacturer accommodates industry's needs. However, these needs may not correspond to the needs of a nuclear power plant protection system.

1.3. Recommendation and Guideline Definition

Suggested guidance is provided in this document beginning with Section 2. Each suggestion consists of a title, motivation or technical basis, and the guidance. The guidance takes the form of either a recommendation or a guideline. For this report, a recommendation is an item that is important to the safety of the system while a guideline is good engineering practice that should be followed to improve the overall quality, reliability, or comprehensibility of the system.

1.4. Structure of This Document

This document provides suggested recommendations and guidelines for PLC applications in nuclear plant protection systems. The topics should be addressed by

the design team in the same order that they appear below.

Section 2 begins with project and configuration management issues. Section 3 addresses the safety-specific concerns involved in the project. Section 4 covers the guidance necessary to ensure that the PLC platform is qualified, including the manufacturer's design, development, tests, maintenance, and modification procedures. The PLC platform consists of the PLC hardware, the operating system, and all software used to develop the application software. Basically, the platform is all the items a user can buy "off the shelf" from a PLC manufacturer. Finally, Section 5 covers the activities necessary to configure, test, install, maintain, and modify the PLC configuration for the plant protection system.

Appendices A and B provide background information for readers wishing more information about PLCs. Appendix A describes a generic PLC and highlights the characteristics important to safety. Appendix B addresses the life cycle of selecting, applying and maintaining a PLC system.

1.5. Motivation for This Guidance

Nuclear power plant utilities are upgrading relay-based systems with PLCs. Some utilities are implementing this upgrade without prior NRC review and approval. 10CFR50.59 allows the licensee to make changes in the nuclear power plant facility without NRC review and approval prior to implementation provided the change does not involve an unreviewed safety question. A change is considered an unreviewed safety question if:

(1) The probability of occurrence or the consequence of an accident or malfunction of equipment important to safety previously evaluated in the Safety Analysis Report may be increased.

(2) The possibility for an accident or malfunction of a different type than any previously evaluated in the Safety Analysis Report may be created.

(3) The margin of safety as defined in the basis for any Technical Specification is reduced.

For most changes the probability of item (1) is perceived as reduced, which is acceptable. For a PLC, this probability can not be quantitatively determined with any confidence. Thus the argument for item (1) is subjective.

When changing from a relay-based or solid-state-based system to a PLC system, item (2) is an unreviewed safety question. The software in the PLC presents a different type of failure mode that could not have been evaluated in the aforementioned type of systems. In addition, a single failure in the PLC central processing unit (CPU) has the potential to degrade a larger portion of the system than any single failure of a relay or solid-state device.

Item (3) may be an unreviewed safety question, because the surveillance, limiting conditions of operation, and/or channel definitions defined in the Technical Specifications may not apply to a PLC-based system. For surveillance, a functional test is sufficient for a relay-based system, but a functional test is not adequate to determine if a PLC-based system is functioning properly. PLC-based systems use self-diagnostics, which must be considered when developing surveillance requirements. The limiting condition of operations for a CPU being out of service will be different from that of a relay or solid-state device. The CPU may degrade many functions while the relay or solid state device may degrade only one. If the PLC-based system relies extensively on multiplexing and communication networks, then channel definitions will probably differ from relay-based or solid-state systems.

New advanced light water reactors (ALWRs) are being proposed that incorporate PLC-based control and protection systems. The NRC has chosen to take the lead in proposing the much-needed guidance for the ALWR vendors, utilities, and regulators on the use of PLCs in nuclear power plant protection systems.

The main body of this document provides the experienced engineer with a review of safety issues in PLC-based systems, while Appendices A and B introduce an inexperienced engineer to PLCs and the use of PLCs in safety applications.

1.6. Special Knowledge Required

1.6.1. Acronyms and Abbreviations

BPCS	Basic Process Control System
CMFA	Common-Mode Failure Analysis
CPU	Central Processing Unit
CRC	Cyclic Redundancy Checking
DCS	Digital Control System
EMI	Electromagnetic Interference
ESD	Emergency Shutdown

FMEA	Failure Mode and Effects Analysis
FTA	Fault Tree Analysis
HWCI	Hardware Configuration Item
I/O	Input and Output
LED	Light Emitting Diode
LRC	Longitudinal Redundancy Checking
MDT	Mean Downtime
MTBF	Mean Time Between Failures
MTDF	Mean Time to Diagnose Failure
MTDL	Mean Time to Determine Fault Location
MTRF	Mean Time to Replace a Faulted Component
MTRO	Mean Time to Return to Operable Condition
MTTR	Mean Time To Repair (MTDL + MTRF + MTRO)
PC	Personal Computer
PES	Programmable Electronic System
PID	Proportional/Integral/Derivative
PLC	Programmable Logic Controller
RAM	Random Access Memory
ROM	Read-Only Memory
SCADA	Supervisory Control and Data Acquisition
SFC	Sequential Function Charts
SFTA	Software Fault Tree Analysis
SWCI	Software Configuration Item
TMR	Triple-Modular Redundant
UPS	Uninterruptible Power Source
VRC	Vertical Redundancy Checking

1.6.2. Specialized Terminology

Availability. (4) (General) The fraction of time that the system is actually capable of performing its mission. (5) (software) (A)The degree to which a system or component is operational and accessible when required for use. (B) The ratio of system up-time to the total operating time. (C) The ability of an item to perform its designated function when required for use. (7) (nuclear power generating station) (A) The characteristic of an item expressed by the probability that it will be operational at a randomly selected future instant in time. (B) Relates to the accessibility of information to the operator on a “continuous,” “sequence” or “as called for” basis. IEEE Std. 100-1988 definition.

Checksum. A deterministic function of a file’s or memory’s contents. If a file is copied and the checksum of the copy is different form the original, there has been an error in copying. The checksum value is a sum obtained by adding the bits in a numeral, or group of numerals, usually without regard to meaning, position, or significance. Modified IEEE Std. 100-1988 definition.

Control network. The PLC communication network which connects the PLC processor to the I/O modules.

Cyclic redundancy check. An error-detection that introduces a check sum at the end of a group of characters that constitute a message. CRC is a popular error-checking scheme in communication systems.

Down time. A period of time that a system is unavailable. Downtime does not include the period of time in which the plant is down for maintenance.

Fail-safe. Fail-safe refers to the output action of a control system upon a failure. A fail-safe control system is one whose outputs operate in such a manner as to reduce the risk of accident when a component or circuit failure occurs in the control loops associated with that failure.

Fail-safe faults. A fault that immediately causes the system to go into a safe state or, in a redundant system, a fault which does not prevent proper and safe control of the process.

Fail-to-danger faults. A fault that prevents the control system from responding to hazard warnings, or can cause a hazardous condition.

Failure mode and effects analysis. A systematic procedure for identifying the modes of failure and for evaluating their consequences. The essential function of an FMEA is to consider each major part of the system, how it may fail (the mode of failure), and what the effect of the failure on the system would be (the failure effect) (IEEE-352).

Fault tree analysis. A technique, either qualitative or quantitative, by which failures that can contribute to an undesired event are organized deductively in a logical process and represented pictorially. It is one way to diagram and communicate the information developed in a failure mode and effects analysis (FMEA) (IEEE-352).

Ladder logic diagrams. Drawings which uses standard symbols (e.g., ISA S5.1—Instrument Symbols and Identification) to designate measurement, logic, and control system interconnections.

Ladder logic. Symbolic programming language used on PLCs. The symbols used in the program relate to ladder logic diagram symbols.

Longitudinal redundancy checking. An error-detection scheme that introduces a single checking character at the end of a group of characters that constitute a message.

Macro. A macro resembles a program subroutine in its use but in the implementation, code is actually copied into the program each time the macro is called. Thus there is no transfer of control from one routine to another each time the macro is used.

Processor-to-processor network. A high-speed communication network which connects PLC processors to other PLC processors, computers, main-frames, etc.

Programmable electronic system. Any computer-based system which controls, protects or monitors the operation of plant machinery or various types of equipment via plant sensors and actuators.

Proportional/Integral/Derivative. Process control algorithm typically used for control of flow, pressure, and liquid level.

Plant safety. An acceptable low risk that a system will maintain plant parameters within acceptable limits established for a design basis event.

Reliability. (1) (general) (B) The probability that a device will function without failure over a specified time period or amount of usage. Notes: (1) Definition (B) is most commonly used in engineering applications. In any case where confusion may arise, specify the definition being used. (2) The probability that the system will perform its function over the specified time should be equal to or greater than the reliability. IEEE Std. 100-1988 definition.

Sequential function charts. Object-oriented programming language used to program PLCs.

System network. The PLC communication network which connects peripherals to the PLC processor. The connection is made through the PLC processor module or a special I/O module.

Vertical redundancy checking. An error detection algorithm also known as parity checking.

2. PROJECT MANAGEMENT GUIDANCE RECOMMENDATIONS

2.1. Project Management Plan

An important element in a successful project is good project management. MIL-STD-499A and IEEE-1058.1 outline a technical and managerial process to document and control a project. A comprehensive process needs to be thought out and written before the project begins. Some projects employing PLCs may be relatively small (one PLC), in which case the plan mentioned here may be the overall organizational plan of the site. It is not necessary that a special plan be produced for each project. It is important, however, that not too much informality be used for implementing a PLC system, even if it is a small system.

Recommendation:

The utility should have a written project management plan in force which details the project organization, responsibilities, managerial process, technical process, tasks, schedule, and budget. This plan may be as short as one page for a small project, and as large as several hundred pages for bigger projects.

(See Appendix B, Section 4.1 for a discussion of project management.)

2.2. Configuration Management Plan

All projects, including those containing PLCs, can benefit from good configuration management. Several standards (MIL-STD-483A, MIL-STD-1456A, MIL-STD-1521B, IEEE-828, and IEEE-1042) address this issue. Through implementation of these standards, the decisions and changes made over the life of the project can be well controlled and documented.

Recommendation:

The utility should have a software and hardware configuration management plan documented and in force. This plan should largely conform either to one of the standard systems listed above or another acceptable standard.

(See Appendix B, Section 4.1 for a discussion of configuration management.)

3. SAFETY GUIDANCE RECOMMENDATIONS

3.1. Safety Plan

As it is important to plan the complete life cycle of a system, beginning with the original conception and continuing to decommissioning, it is equally important to define a safety life cycle when safety is an important issue. The safety life cycle model is a part of a safety plan. During the system requirements phase of the project, the safety plan is developed in outline form. Throughout the remainder of the project the safety plan is completed and updated. The life cycle model shows the phases of the project that require safety specific activities. The IEC is developing a standard (IEC-65A) that addresses safety issues for programmable systems, and specifically addresses the safety plan and the safety life cycle model.

Recommendation:

The utility should generate documents detailing the design requirements from a safety perspective. Also, a safety life cycle model needs to be developed as part of the overall system life cycle.

3.2. Features Required for Safe Operation

Specific features of the system that are required for safe operation must be clearly delineated. To help focus on safety and provide a decision path for others, the reason each feature is important to safety should be stated. Some features which should be considered for PLC systems are:

- Module “hot swapping.”
- RAM battery back-up and battery monitor.
- System battery back-up.
- Redundancy.
- Fault tolerance.
- Acceptable system availability.
- Clearly defined fail-safe modes.

Recommendation:

Specific features of the system that are required for safe operation are to be identified and the reason they are important to safety needs to be documented.

(See Appendix A for additional information on PLC features.)

3.3. Hazard and Risk Analysis

A safety analysis should be performed that identifies the hazards and risks that the system will have to mitigate. Once the hazards and risks are identified the safety function can be defined. The system response to each hazard and risk must be clearly specified. Document IEC-65A discusses the objectives and requirements of a hazard and risk analysis.

Recommendation:

The hazards and risks identified and the safety functions that are necessary to mitigate them need to be specified. The hazard and risk analysis identifies the hazards, defines the event sequence leading up to each hazard, and determines the risk associated with each hazard. For each anticipated hazard or risk the system’s response should be described and found acceptable.

(See Appendix B, Section 4.2 for a discussion of safety analysis.)

3.4. Failure Analysis

Failure analysis is an activity that will be completed before the selection of the ESD system (to establish that the system can meet the performance goals) and verified after engineering and installation. In the system requirements phase of the project, an outline of the Safety Plan should be developed, and that outline should call out the need for this activity.

An important exercise in producing a safe system is to analyze the system to find unsafe states. By systematically identifying these states, the system designers should be able to re-design in order to reduce the number of unsafe states and prove acceptability of any remaining unsafe states. Some potential problems to be considered are:

- Power transients, excursions, and dips.
- Auto-recoveries from power loss.
- Initialization routines for PLC system start-up.
- Shutdown routines for the PLC system.
- Memory loss or corruption.
- Communication loss or corruption.
- I/O module failures.

- Unreadable or unread inputs.
- Addressing errors.
- Processor faults, both in the PLC CPU and the I/O modules' CPU.

The FMEA and FTA (IEEE-352) are tools that can be used to systematically analyze failures of a system and the effects of those failures. Common-cause failures are less frequent than single failures (which are analyzed in an FMEA or FTA) but can be much more severe. EPRI has written a document, EPRI NP-5613, which addresses the issues of adding common-cause failure effects into an FMEA or FTA.

Recommendation:

Failures that put the system into unsafe states are to be determined, and ways to reduce the number of these unsafe states should be explored. Single failures and common-cause failures should be analyzed.

(See Appendix B, Section 4.2 for a discussion of safety analysis.)

3.5. Quantification of System Reliability

Quantification of system reliability is an activity that will be completed before the selection of the ESD system (to establish that the system can meet the performance goals) and verified after engineering and installation. In the system requirements phase of the project, an outline of the Safety Plan should be developed, and that outline should call out the need for this activity.

Various reliability analysis techniques are available. Both qualitative and quantitative methods can be used to gain a better understanding of the system and a higher confidence that the system will perform its required functions.

Many parametric modeling techniques are available and can be used to quantify the reliability of hardware. One group in particular, ISA Subcommittee SP84.02, has done extensive work modeling PLC systems and quantifying the reliability of the hardware. This committee has written a document, ISA SP84.02, which uses Markov models to quantify the reliability of 14 different PLC architectures. EPRI document EPRI NP-5613 references other parametric modeling techniques that can be used to quantify reliability of hardware systems.

A PLC system can be thought of as having two major components—software and hardware. The techniques mentioned above are easily applied to hardware, but characterizing reliability for software is much more difficult. The analyses mentioned here typically will not include software unless there is a solid statistical basis for quantifying the software reliability.

Guideline:

A parametric model of the PLC system should be developed and used to quantify the probability of system failure.

(See Appendix B, Section 4.2 for a discussion of safety analysis.)

3.6. Quantification of System Availability

Quantification of system availability is an activity that will be completed before the selection of the ESD system (to establish that the system can meet the performance goals) and verified after engineering and installation. In the system requirements phase of the project an outline of the Safety Plan should be developed, and that outline should call out the need for this activity.

Calculating an availability number along with the estimated hazard demand on the system can be valuable in estimating system safety. Typically, the influence of software on availability is not factored in because of the difficulty of quantifying software reliability.

Guideline:

The method of calculating the desired availability of the system should be clearly documented. Estimates of the various constituents of availability need to be made and documented. In addition, the desired system availability needs to be determined. The acceptable availability will depend heavily on the desired hazard rate. It is important to document all steps performed and decisions made to reach the availability number. Also, any changes should be documented, along with reasons for making the changes.

3.7. Quantification of System Hazard Rate

Quantification of system hazard rate is an activity that will be completed before the selection of the ESD system. At the system requirements phase of the project an outline of the Safety Plan should be developed, and that outline should call out the need for this activity.

Hazard rate is a measure that relates the system availability and reliability to the concerns of safety. How low the hazard rate should be can only be determined by acceptable industry standards, the people at risk, and the plant managers. Of course, the hazard rate can not equal zero, but neither should the hazard rate be entirely unacceptable by any of the groups mentioned above.

Guideline:

The process of determining the acceptable hazard rate and the means to achieve it should be clearly indicated in the documentation of the safety analysis.

(See Appendix B, Section 4.2 for a discussion of safety analysis.)

3.8. Software Reliability Issues

Determination of software reliability should be completed after software has been written. In the system requirements phase of the project an outline of the Safety Plan should be developed, and that outline should call out the need for this activity.

The techniques discussed above can easily accommodate hardware, but applying them to software is a much more difficult task and they are rarely, if ever, used for this purpose. Nancy Leveson of U.C. Irvine has been doing some work with software fault tree analysis (SFTA). She presents a procedure based on FTA techniques to improve the reliability and safety of the code. Although FTA is quantitative, SFTA is qualitative. The SFTA breaks each hazard down to software components. Once these components are identified, the developer can act to reduce the number of hazards. More detailed information on SFTA and examples of its use can be found in Leveson and Harvey 1983 and Leveson et al 1991.

Formal Methods can improve software reliability and are getting much attention in the software community. Ladder logic lends itself to Formal Methods. The approach of Formal Methods is to derive mathematical equations for the system outputs in terms of the system inputs. Ladder logic programming (assuming no complex function blocks are used) is convertible into Boolean equations. Thus, if the system requirements are defined with Boolean equations, a comparison of system level Boolean equations and ladder logic program Boolean equations can be made.

Guideline:

It is useful to express the ladder logic to be implemented in Boolean equation form prior to implementation. This could be the language of the requirements document. After implementation the equations could be re-derived from the ladder logic, taking into account the idiosyncrasies of the implementation that the PLC imposes. As a check, the Boolean equations derived from the ladder diagrams can be compared to the equations used to specify the ladders. The person deriving the equations from the ladder should be different from the person who performs the implementation from the equations and from the person who produced the equations in the first place.

If other programming languages are used (C, for example), the standard methods for software verification and validation (V&V) should be employed, together with good programming practice (such as inspections).

(See Appendix B, Section 4.3.2 for further discussion of software concerns.)

4. PLC QUALIFICATION GUIDANCE RECOMMENDATIONS

Section 4 deals with qualification of the manufacturer's hardware and software. It specifically excludes all application software which may be written in Ladder Logic language, state-based language, Sequential Function Chart language, Boolean language, configuration tables, or any other configurable system language. Protection system application software for the PLC system that is written in C, BASIC, or any other common source language is also not treated in this section.

4.1. Hardware Qualification

4.1.1. Environmental and Class 1E Requirements

PLCs must be able to perform reliably in the environment in which they will be installed.

Recommendation:

A PLC system for use in a reactor protection system should be qualified as a Class 1E system.

4.1.2. Communication Systems

Most PLC systems can communicate over many types of media with various communication protocols. Any system that communicates with the PLC system can cause the PLC system to "lock up." A partial list includes devices such as printers and cassette loaders, as well as systems using one-way messages to other systems that incorporate "hand-shaking" or error correction techniques. The issue of reliable communications is a complex topic and requires a detailed understanding of the communication systems and understanding of the safety implications imposed on the protection system via communication failures. Preckshot 1993b discusses the current state of digital communication systems. In addition, Preckshot 1993b provides recommendations and guidance on communication systems for nuclear power plant applications.

Recommendation:

All communication systems in the PLC-based PES should adhere to the recommendations and follow the guidelines of Preckshot 1993b.

(See Appendix A, Section 7 for a discussion of PLC communications.)

4.1.3. Downloading Configurations to I/O Modules

The current trend in configurable electronics is to have the host device (e.g. the PLC CPU or a programming terminal) download configurations to I/O modules. One potential problem that must be addressed in this case is the question of whether the configuration will survive a power failure. If it will not, then the module should have a means of indicating the configuration loss to the PLC. The PLC user program must then sense the module's memory loss and reload the configuration. It should be noted that I/O modules are usually located remotely from the PLC and are not necessarily subject to the same power losses to which the PLC is subject. Therefore, it is not sufficient to monitor only power loss at the PLC itself.

Recommendation:

Special consideration should be given to PLC systems having I/O modules that store configuration information in RAM. The hazards and risk of losing data from or having corrupted data in the I/O module's RAM must be evaluated and documented in the Safety Plan. In addition, other safety considerations such as power loss and automatic re-start must be addressed in the Safety Plan.

(See Appendix A, Section 8.1 for more information on programming concerns.)

4.1.4. Battery Back-Up of RAM

Most PLC systems contain volatile RAM in the PLC CPU and there also may be volatile RAM in various I/O modules. This RAM loses its memory upon loss of power. In order to retain memory through system power outages, PLC systems have battery back-up. Typically, the battery's life will be several years.

To facilitate maintenance of these batteries, most manufacturers provide indicators to warn the operators when a battery is near failure. Typically, the PLC system is in a cabinet and the only way to see a local indicator is to open the cabinet doors. It is much better if the PLC manufacturer provides the capability to remotely indicate the low-battery status. If I/O modules have a backup battery, then the module should annunciate their impending failure to the PLC, which will then take the appropriate action.

Recommendation:

The PLC system should have a means to report to the operators and maintenance personnel low battery power on all RAM back-up batteries. Administrative procedures and checklists are the minimum necessary for monitoring battery life.

(See Appendix A, Section 6 for more information on power supplies.)

4.1.5. Circuit Protection on Output Modules

Most output modules that switch power to field devices have equipment such as fuses or circuit breakers to protect the output electronics. When these circuit protectors trip they eliminate the output module's capability to control the field device.

Recommendation:

All I/O points with circuit protection should be under surveillance.

(See Appendix A, Section 5 for more information on output structures.)

4.1.6. I/O Module Terminations

I/O module terminations provide the interconnection between the field input/output devices and the PLC system. These terminations are available in a variety of designs and configurations. To maintain the integrity of the connections over the life of the equipment, (1) the I/O modules of the PLC system should be removable without disturbing the field wiring connections, and (2) there should be a mechanism to ensure that the field connections will not loosen over the life of the equipment.

Guideline:

The PLC system should allow I/O module removal without disconnecting any of the field wiring. Further, the field wiring terminations should lock down in a manner that will not allow the wires to loosen over the life of the equipment.

(See Appendix A, Sections 4 and 5 for more information on input and output structures.)

4.2. Software Qualification

Finding methods to prove the safety and reliability of software is an elusive goal that is being sought by many people in the software community.

The software provided by the PLC manufacturer should meet all of the standards applying to software used in the intended application, which requires the purchaser of the software to inspect the process that the PLC vendor uses in the production of the software to verify that there is a software management plan, a configuration management plan, a QA plan, a V&V plan, etc., all in place, in force and being executed. Further, these plans should be equivalent to those that would be required for any software produced and used in the intended application.

A weak alternative to the above is the process of "commercial dedication," in which a software vendor has enough experience with the software in actual use to justify the assertion that the software is sufficiently reliable for the intended application. This experience should include a method for receiving problem reports from the field and then resolving them through modifications when appropriate. There should also be a method for proving that changes, when made, in fact improve the system and do not install other errors. This may be determinable through regression testing. Note that an organization that can successfully handle this task (resolving problems) probably has a reasonably good method for developing software in the first place.

The best evidence of qualification is that both of the above alternatives are available.

Recommendation:

PLC manufacturers should be able to demonstrate to the purchaser that they are using good software engineering practice to produce their software, and that the software is sufficiently reliable to meet the requirements of the intended application. Alternatively, they should be able to show that there is adequate experience in actual operation of the proposed software in the field to demonstrate that the reliability of the software is adequate for the intended application.

(See Appendix B, Section 4.3 for further discussion of PLC considerations.)

5. PLC APPLICATION GUIDANCE RECOMMENDATIONS

5.1. PLC Configuration

Section 5 addresses the configuration of the PLC hardware and software. The software may be written in Ladder Logic, state machine language, Sequential Function Chart language, Boolean language, configuration tables, or other configurable system language. Application software written in C, BASIC, or other common source language is also addressed in this section.

5.1.1. Real-Time Performance

Real-time performance is of utmost importance for a protection system. The system must be able to respond to a hazard within a specified time. A worst-case timing analysis would be appropriate after the hardware configuration is complete and before implementation. This analysis, followed by a timing test, could prove the system meets all the real-time requirements. Lawrence 1992b address real-time performance issues relative to protection system applications.

Recommendation:

The real-time performance requirements of the system should adhere to the recommendations and follow the guidelines of Lawrence 1992b.

5.1.2. Formal Configuration Process

Software development, regardless of the language used, benefits from the application of a formal process. A requirements document should be produced that contains all of the requirements the system must meet. For instance, if ladder logic is to be the language of implementation, Boolean equations provide a clear, unambiguous method of specifying what must be accomplished, but Boolean equations alone are inadequate. Clear, natural-language specifications should be included, whatever method is used, in order that details of the requirements can be made clear to future readers. Each requirement should be testable so that the implementation can be validated when completed. Untestable statements in the requirements document are not requirements and only add confusion.

Once the requirements documents have been inspected and approved by an appropriate authority, design may proceed. This inspection and approval process corresponds to requirements verification in the more conventional software world. For PLC applications, design and implementation may blend together in such a way that the two operations become inseparable, yet there should be a document that contains the complete design, even if it looks like an implementation. Relying on a soft copy in the implementing machine is inadequate. Further, a document is needed that maps the input and output modules to the field devices to which the modules are connected.

Next, appropriate reviews and inspections of the design and implementation must be performed in order to eliminate errors before starting the test activity. These reviews and inspections correspond to independent verification of the software. Reports should be issued to document the inspections and reviews performed, deficiencies found and remedial action taken.

Prior to installation, the system should be “rung out” and tested to verify that the system meets all of its requirements. This step corresponds to software validation in more conventional software systems. Test plans and procedures need to be written and test results documented showing any deficiencies discovered and remedial action taken.

When the installation is complete, an integration test should be run to demonstrate that the connections are all correct and that the installation is complete. Test plans and procedures need to be written and test results documented showing any deficiencies discovered and remedial action taken.

Finally, all of the above mentioned documents should be updated to represent the system as built, and they should all be put under configuration control.

The process described above applies to systems consisting of one or two PLCs connected to some hardware. As more complex systems are contemplated, more formal methods should be applied, up to the point at which a full-blown project management system is in place, including a full formal software development plan and life cycle. The auditor should tailor his audit to the complexity of the system. Simple systems should require simple plans and simple documentation, while complex systems should be planned and documented accordingly.

Recommendation:

Plans and documentation need to be produced that are appropriate to the size of the system being built. These items should be complete, accurate, and should clearly indicate that a process adequate for the application was used in the production of the software.

(See Appendix B for a discussion of PLC considerations.)

5.1.3. Software Modularization

In many cases, certain functions appear repeatedly in an implementation. For example, circuits to start and stop motors may appear several times in a system. For such functions, standardized configurations should be produced and used whenever appropriate. Modularization of the software configuration makes the software easier to read, understand, and test.

Guideline:

To the extent possible, the configuration of the software should be modularized into functional blocks.

(See Appendix A, Sections 4 and 5 for more information on input and output structures.)

5.1.4. Common Programming Languages

Many manufacturers are providing the ability to program their PLCs or their I/O modules with common programming languages (C, BASIC, FORTRAN, etc.) or some dialect of these languages. The manufacturer provides all the necessary features to incorporate a user-programmed subroutine. For ladder logic a “program block” is provided in which a user-programmed subroutine can be written. The subroutine has direct control over the PLC processor’s registers and performs read, write, and computational operations. The program block has an input to start the subroutine and status outputs that can be used to control other ladder logic instructions. One of the advantages of a PLC over a more conventional computer system is that the programming language is a very-high-level language aimed at a specific application (e.g., ladder logic). The introduction of conventional computer languages compromises that simplicity. Another advantage of the PLC executing only ladder logic is that the program scan is deterministic. With the introduction of program blocks, the system may become non-deterministic. Further, program blocks may allow the user to read or write to any RAM location in the PLC. Typically, the programmer only manipulates the PLC processor registers, but it may be possible to manipulate vital areas of RAM such as the configuration data, I/O mapping data, or I/O image tables. Thus, using common programming languages in a PLC allows the programmer to completely subvert the operating system and the interpreter with, perhaps, disastrous consequences.

Recommendation:

When common programming languages are used in the PLC system, documentation and development methods shall be employed that are adequate for the safety criticality of the application.

(See IEC-880, IEEE/ANS P-7-4.3.2, or Preckshot 1993c for more information.)

5.1.5. Complex Instructions

Certain instructions can be misused or their effects can be overlooked, due to the complexity they add to the software. Ladder logic instructions such as JUMP, GOTO, SKIP, or interrupts (e.g., Processor Input Interrupts—this instruction/function interrupts the PLC CPU and executes a ladder logic subroutine) are examples. This is a difficult area because the subtle effects of these instructions may be overlooked. Thus, problems that may occur must be considered while employing these instructions. The possibility of an error occurring is particularly likely over the long term. For instance, the original programmer may use a subtle instruction without documenting it carefully enough, and then a later programmer making a modification causes a system failure because he did not recognize the effect of the subtlety. Further, each PLC vendor provides a different set of commands, which further compounds the problem of added complexity.

Recommendation:

The implementation should be as simple as possible, avoiding complex data structures, program blocks, instructions with concealed side effects, etc. It should be possible to look at a sample of the code to see if such elements are employed.

(See Appendix A, Section 3.4.2 for more information on complex functions.)

5.1.6. Remote Operations

Some PLCs allow control of the software configuration via remote computers using communication links. The remote computer can change the PLC configuration, force outputs to particular states, and change the mode of the PLC (e.g. from “run” to “stop”). If this type of operation is allowed at all, it should be under strict control.

Recommendation:

Remote operation of any PLC should be allowed only with written approval, for a strictly limited and documented period of time, for a specific purpose spelled out in the approval document, and under strict supervision.

(See Appendix A, Section 8.1 for more information on remote operations.)

5.1.7. Software Style

When writing software for safety-related applications, considerable effort should be expended to produce code that is as clear as possible. Writing compact and “efficient” code should be dispensed with if it interferes with clarity. This effort is particularly important for nuclear systems, whose lifetimes are several decades. It is unreasonable to expect that the programmer who originally programmed an application will be present to maintain it near the end of its life. The several people who have responsibility for the code over its life need to have a code that can be easily understood.

Guideline:

Every attempt should be made to configure the software in a manner that is easily understood. Understandability of the configuration should be at least as high a priority as correctness of function.

(See Appendix B, Section 4.3.2 for more information on PLC software considerations.)

5.1.8. Latched Outputs

A common practice for starting equipment is to send a pulse to start the equipment, and “seal in” the pulse to keep the equipment running. The advantage of a seal-in circuit is that when power is cycled off and then on, the equipment shuts down and stays shut down until another start pulse is received. Sealed-in circuits eliminate the inadvertent re-start of motors, pumps, etc. when system power cycles.

Recommendation:

For latched outputs that start equipment, the programmer should not program latched outputs that stay latched through power cycling. A holding circuit with “seal-in” contacts, commonly done with relays, should be implemented in the PLC program.

(See Appendix B, Section 4.3.2 for more information on PLC software considerations.)

5.1.9. “Queue Full/Empty” Indicators

Some PLCs make queue data structures available to the programmer. These queues can be either LIFO (Last In First Out) or FIFO (First In First Out) structures. Most manufacturers provide status bits for the queue operation. Typically, the status bits are *enable*, *full*, and *empty*. The queues used may have full and empty indicators. If so, the programmer should use these indicators to ensure that the data being loaded or unloaded from the queue is not lost or invalid (i.e., the programmer should take appropriate actions if a full or empty status is determined).

Guideline:

PLCs with no queue full or empty indicators are not recommended for use in safety shutdown applications. For software configurations in which a queue is used it is the PLC programmer’s responsibility to test for the queue status bits and take appropriate action.

(See Appendix A, Section 3.4.2 for a discussion of complex functions.)

5.1.10. Error Handling

Errors can occur occasionally. For example, if arithmetic is being done in a function block, divide-by-zero will cause an error condition to be asserted and the function block to be terminated. If this error condition is not handled by the programmer, the action of the program may be unexpected and lead to a hazardous condition.

Guideline:

The software configuration should test for error conditions and take appropriate action.

(See Appendix A, Section 3.4.2 for a discussion of complex functions.)

5.2. Configuration Testing

Typically, design and development of the hardware and software will progress in parallel. After completion of the hardware configuration and field connections, the hardware portion of the safety system should be exercised without the safety system software configuration installed. The recommendation and guidelines for testing the hardware portion of the safety system are detailed below. Items 5.2.1 and 5.2.2 apply to testing of both the hardware and software configuration. The remaining items address testing and V&V issues associated with configuration of the software. In these software testing items, it is assumed that the hardware portions of the safety system have passed all of their functional testing.

5.2.1. Test Plan

The components of the system, the subsystems, and the complete system need to be tested. These tests may occur at various sites and be performed by various people. The key is not by whom or in what location the test is completed, but that the test is well thought out, completed, documented and independently verified. In order to maximize the areas of the system that are exercised in the limited time available for testing, a well-thought-out plan should be developed before testing begins. The test documentation should provide an auditor with an understanding of what was tested, why it was tested, and how much of the system was tested. ISA's manual, ISA RP55.1 provides a comprehensive discussion on testing process computers.

Recommendation:

A test plan should be written that clearly describes all steps in the test, defines what constitutes success, and details how the test results will be documented.

(See Appendix B, Section 4.4 for more information on testing considerations.)

5.2.2. Failure Documentation

Documenting failures and calculating statistics can be invaluable. Keeping accurate historical records of the failures can show trends in component failures. These trends can help determine failure-prone areas in the system. By keeping complete records of all failures that occur in the system's life, the system's expected reliability can be calculated. This process of collecting statistics and keeping records on failures starts during testing and continues for the life of the system.

Recommendation:

All failures and appropriate parameters (time of occurrence, time to repair, etc.) that occur during testing should be recorded. For each failure, the reason for the failure and corrective action taken should be recorded.

(See Appendix B, Section 4.6 for more information on hardware and software maintenance.)

5.2.3. Software Verification and Validation

Software verification and validation (V&V) is an integral part of the software life cycle and relates to the safety life cycle. Proper implementation of a verification and validation process can improve the quality of the software and the configuration's adherence to the original system and safety requirements.

Recommendation:

There should be a formal independent verification and validation process for all of the PLC application software.

(See IEEE P-7-4.3.2 for a discussion of software V&V.)

5.2.4. Software Fault Tree Analysis

SFTA is a process used to improve the safety of the software by decomposing the code into root causes of postulated failures. SFTA is an extension of the more popular FTA performed on electromechanical systems that includes the software components within the FTA. In SFTA, hazards are identified and decomposed into the events leading to the hazard. The events can be hardware failures, software failures, human errors, etc. The goal is to reduce the number of root causes. The PLC configurable software then can be reorganized or re-configured to eliminate the software branches or the software-based root causes in the fault tree. It is inconceivable that all the root causes, or even all the software root causes will be eliminated. Most programmers develop the software configuration from the point of view of what they want the software to do.

SFTA looks at what the software shall not do. The SFTA starts with the system fault tree, which is different from the software requirements specification and thus can expose errors in the software requirements specification. By providing an alternate view of the software and starting from a different specification, the SFTA results in a greater confidence that the software will mitigate a hazard. Reports by Bowman et al., Leveson and Harvey 1983, and Leveson et al 1991 illustrate the software FTA process and examples of its use.

Guideline:

Perform a Software Fault Tree Analysis on the PLC configurable software as detailed in the papers mentioned above. The SFTA is a tool to help expose errors in the software configuration and the software requirements. As such, the level of analysis will depend on the complexity of the software configuration.

(See Leveson et al 1991 for more information on fault tree analysis.)

5.3. Installation

5.3.1. Installation Practice

Proper installation of the system is critical to reliable operation. Improper installation can cause failures that may have severe consequences. Good installation techniques will make maintaining the system easier and reduce the possibility of maintenance errors.

Guideline:

Verify that (1) correct hardware and software configurations are properly installed, (2) good industry practice is being used in wiring, (3) the proper versions of the software (i.e., operating system, translation software, tools, software configuration, etc.) are installed, and (4) various pieces of the system are installed in the correct environmental conditions.

(See Appendix B, Section 4.5 for more information on installation concerns.)

5.3.2. Parallel Output Devices

Output devices should not be connected in parallel to increase current carrying capacity. Output device specifications will not guarantee simultaneous switching for identical devices. Thus, if two identical devices are connected in parallel, the first one to turn on will be excessively stressed. Also, the “ON” resistance may be different in each device, causing the current division to be unequal. This may also cause excessive stress in one of the devices. Connecting output devices in parallel to increase current capacity causes an abnormally high rate of failures (Wilhelm 1985). The proper design will use one switching device with the proper interface. If parallel output devices are desired for a fail-safe design, then each output device should be rated to properly handle the full load.

Recommendation:

No output devices should be connected in parallel to increase current carrying capacity. Each output device should be properly rated to handle the full load expected from the field device.

(See Appendix B, Section 4.5 for more information on installation concerns.)

5.3.3. Inductive Voltage Spikes

An inductive DC load connected to an output module should have a diode in parallel with the load or other mechanism to suppress the surge when the inductor is switched off. This surge can create a voltage spike if there is no suppression mechanism. This spike may damage the output module.

Recommendation:

An inductive DC load should have a surge suppression mechanism installed. This mechanism may be supplied in the output module by the manufacturer or may be installed as part of the field wiring.

(See Appendix B, Section 4.5 for more information on installation concerns.)

5.3.4. Power Source Isolation

Computers are sensitive to power source variations. Voltage or current surges in the power source can cause damage to the PLC electronics.

Recommendation:

The PLC system should have its own isolated power source and, if the system requires it, an uninterruptible power source (UPS).

(See Appendix B, Section 4.5 for more information on installation concerns.)

5.4. Maintenance

5.4.1. Technical Specification Update

With the introduction of a PLC into the system, some new items may have to be included in the technical specifications for the plant. For example, the back-up batteries will need to have surveillance performed from time to time, as will any air filters on the air intakes of the PLC cabinets.

Recommendation:

The technical specifications of the plant need to be updated to include all applicable items from the PLC system.

(See Appendix B, Section 4.6 for more information on maintenance concerns.)

5.4.2. Documentation of Failures

Tracking failures can help operators to forecast trends in high-failure-rate components or identify software configurations with high failure rates. Tracking can help determine which hardware components should be kept on the shelf as spare parts. High failure rates on a certain part of the system can indicate a problem, which then may be investigated.

Guideline:

When a failure occurs the system should be thoroughly diagnosed and all hardware problems found and repaired. There should be in place and in force a system for documenting all hardware failures, software failures, software errors, and design errors. The documentation should include when the failure or error occurred, what diagnostics were run, what the diagnostic discovered, the resolution, and what was replaced or changed.

(See Appendix B, Section 4.6 for more information on maintenance concerns.)

5.4.3. PLC Module Replacement

Typically, maintenance of a PLC system is done by removing and replacing modules. Some PLC systems allow this to occur without removing power from the module to be replaced, but some do not. Maintenance procedures should clearly spell out what procedures are to be followed when a module is to be replaced. Further, some modules require customization prior to installation. If such is the case, those modules requiring customization should be clearly identified on the module, in order that the proper customization is not forgotten when an installation is anticipated.

Most PLC vendors provide mechanisms that prevent the installation of an incorrect module in a slot. One such mechanism is a slot key, which is customized for the module to be installed in the slot, and therefore prevents the installation of the wrong module. Another mechanism is the “traffic cop”—software that interrogates a module after installation to see if it is the correct one and prints an error message if it is not. (In such systems plugging in the wrong module will not damage the module but it will prevent correct operation.)

Recommendation:

Mechanical, electrical or administrative controls should be in place so that when modules are changed out for maintenance, modules are not damaged and the correct module is properly configured and installed.

(See Appendix B, Section 4.6.1 for more information on hardware maintenance concerns.)

5.4.4. Input/Output Forcing

“Forcing” is the action of putting a binary PLC input or output into a known state. In most PLC systems, a user can force any of the I/O points. This operation is widely used in debugging the program and verifying proper operation of I/O points and connected equipment. However, considerable care should be exercised when using this function. Forcing after installation causes equipment actuation (valves to close/open, motors to start/stop, etc.). Devices such as safety interlocks, motor hold-in contacts, and other safety circuits can be bypassed or modified by forced conditions, increasing the probability of a hazard occurring. In addition, once an I/O point is forced, the PLC may not automatically return the I/O point to normal operation. Most PLC systems remove all forced I/O points once the programming terminal is switched off or disconnected from the PLC. The user should know enough about the plant process, the PLC system, and the characteristics of the forcing command to use forcing properly. Further, in safety applications, the forcing commands should not be used without strict administrative control.

Recommendation:

There should be a means to control the use of the "forcing" function. Forced conditions should only be allowed under strict administrative control when the PLC system is providing protection during plant operation.

(See Appendix B, Section 4.6.2 for more information on software maintenance concerns.)

5.5. Modifications

5.5.1. Modifications

Modifications to the hardware or software may be requested by engineering, operations, maintenance, the PLC manufacturer, and others. Modifications may be enhancements to the system or fixes to problems. A formal procedure should be in place to receive problem reports, review them, suggest modifications to fix the problem, and to implement and test the modification when a modification is appropriate. Further, this procedure should include the assessment of modifications provided by the PLC vendor prior to installation. Any modification, especially a software modification, has the potential to introduce hazardous states into the system.

A modification should be checked for correct operation under all plausible operational scenarios. Interactions with other equipment should be checked for proper operation. And finally, after a modification is installed, it should be closely monitored and if problems arise, it should be possible to revert to the old system while the problems are considered.

Recommendation:

The utility should have a formal procedure to report, review, implement, and test all modifications. Unauthorized modifications should not be allowed.

(See Appendix B, Section 4.7 for more information on software modifications.)

5.5.2. On-Line Programming

On-line programming allows the user to change the PLC program or data while the PLC is in run mode (operational). When using this feature, the user should be very careful not to cause undesirable actions in the control process. In safety-critical applications, tight control should be maintained on all devices that allow on-line programming. The personnel making a change should fully understand the process, the equipment being controlled, and the operation of the PLC system.

Recommendation:

There should be an administrative procedure to control the use of the on-line programming feature. On-line programming should only be allowed under strict administrative control when the PLC system is providing protection during plant operation.

(See Appendix B, Section 4.7.2 for more information on software modifications.)

Appendix A: Programmable Logic Controller Characteristics and Safety Considerations

J. Palomar

R. Wyman

Manuscript Date: June 30, 1993

Contents

Figures

Appendix A: Programmable Logic Controller Characteristics and Safety Considerations

1. INTRODUCTION

Appendix A provides an overview of programmable logic controller (PLC) characteristics. The intent is to give the reader some understanding of the most popular characteristics of programmable logic controllers. No attempt is made to discuss every characteristic available in the industry; however, characteristics that make the programmable logic controller more suitable for emergency shutdown than other electrical/electronic based systems or improve reliability are described in more detail. In addition, characteristics that may provide an unsafe operating environment are also commented on.

2. PLC DEFINITION

Since the advent of the microprocessor, digital systems have been taking over more real-time control system functions. Digital control systems have become an accepted standard for control. Digital control systems capture and utilize the power of the microprocessor. This power has been put to good use in control of all types of processes, from small research and development systems running a couple of motors to huge control systems running oil refineries, steel mills, and power plants. Commercial computers such as the IBM PC, Macintosh, DEC computers, and Hewlett Packard computers are general-purpose computers. Special-purpose computer systems have been designed to improve the performance, user interface, and operation of various control schemes. Such designs are often referred to as digital controllers, and they have been designed for specific applications such as motor control, PID control, and sequencing logic.

One such special purpose computer or digital controller, originally designed to replace industrial relay-based control systems, is the Programmable Logic Controller (PLC). The PLC has evolved into more than a logic solver and with this has come an alternate name of programmable controller, PC. To eliminate the confusion between personnel computer, PC and programmable controller, PC, this paper will refer to a programmable controller or a programmable logic controller as a PLC. The major difference between a PC and a PLC is the programming language. The common programming languages for a PLC are relay ladder programs, Boolean equations, and sequential function charts.

There are very few differences between a PLC and other digital control systems. NEMA ICS3-1978, Part ICS3-304, defines a PLC in general terms that include computers and digital controllers of all types. For the purposes of this paper, a more focused definition of the PLC is as follows:

A programmable logic controller is a digital operating electronic apparatus which uses a programmable memory for the internal storage of instruction for implementing specific functions such as logic, sequencing, timing, counting, and arithmetic to control, through digital or analog input/output modules, various type of machines or processes. The digital apparatus must offer at least one restrictive, higher level, programming language such as ladder logic programming, Boolean programming, or sequential function charts; must contain an operating system that can execute its software in a deterministic manner; and must interface primarily to sensors and actuating devices. The programming language must offer as a minimum relay coil and contact, timing, counting, and latch instructions.

3. INTELLIGENCE

Many features of the computer system may be described in human terms, such as the CPU being the “brains” of the system. In this paper, the program and

the PLC operating system provide what will be called machine intelligence.

The machine intelligence of the PLC is derived from microprocessor-based electronics. At a minimum, a PLC system consists of a central processing unit (CPU), read-only memory (ROM), random access memory (RAM), programming terminal interface electronics, and I/O interfacing electronics. The CPU handles all activities of the PLC system. The CPU provides a user programming environment, executes the user program, analyzes incoming data, and responds to the incoming data via control signals to the output modules. Every PLC offers basic relay functions, and most expand the functionality to cover a wide variety of complex functions.

3.1. CPU

All PLCs contain at least one CPU, (typically one electronic printed circuit board) that executes user program instructions. It is the central unit that guides all operation within the PLC. In more complex systems, this unit will communicate with and control the operation of other subsystems within the PLC. Other subsystems may be arithmetic logic units, floating point processors or co-processors. However, the distinguishing feature of the CPU is that it has central control over the entire PLC system. The subsystems may contain microprocessors, but their control is limited to the subsystem.

3.2. Memory

Two basic types of memory are available to the CPU—ROM and RAM. Typically, the operating system and programming language commands are stored in ROM, while the user program and the input/output data are stored in RAM. ROM memory cannot be changed by the CPU, while RAM memory can be.

ROM memory is programmed at the time of manufacture, and the only way to change the ROM program is by replacing the ROM hardware. Other types of ROM exist which may be programmed after manufacture, called as programmable read-only memory or PROM. They are erasable by high voltage (25–50 Vdc) or ultraviolet light, and can be re-programmed after erasure. The aforementioned types of ROM must be removed from the circuit in order to be reprogrammed, but a newer type, called electrically alterable read-only memory (EAROM), may be erased and re-programmed while in the circuit.

RAM memory may be changed many times by the CPU. The CPU uses the RAM to store the constantly changing input/output data, intermediate calculations, user programs, and various data that must change during the operation of the PLC. Two types of RAM are available, non-volatile and volatile. Non-volatile memory holds its memory values even if power is removed. This is known as core memory and utilizes magnetic fields to store bit states. Much more common is volatile memory made of semiconductor material. This type of RAM requires battery back-up power to sustain memory through a power outage. This is a key maintenance issue that should be addressed before selection of a PLC system. The PLC system should have a means to indicate low battery power to the users. Local indicators on the PLC are not sufficient. The battery low status should be reported to all user interface devices as an alarm condition.

3.3. Operation

The PLC has various modes of operation. One mode, Run-Time, is of particular importance, and an understanding of this mode will aid in the understanding of the following sections. Run-Time specifies the period of time in which the PLC executes the user’s program. It can be thought of as a sequential process with five major steps. The first step is to scan all input modules, including any error checking of addresses/data and diagnostics. Next, an input image table in the PLC RAM is updated. Third, the CPU executes the user’s logic program step by step, line by line. The data in the input image table is used as needed in each step. Fourth, the output image table can be updated. Finally, the PLC outputs the results to the modules. The execution of all five steps is called a scan cycle. Note that the entire user program is executed in one scan cycle. The PLC repeats the scan cycle until it is stopped by the user or shut down in some other way. The five steps help visualize the flow of data, although in actual implementation, the execution of the steps may overlap in time. Figure A-1 outlines the Run-Time steps.

3.4. Functionality

The PLC was designed to replace systems of industrial control relays. The PLC increased the reliability, increased the control information and data available to the operators, and decreased the effort involved to retrofit a relay-based system. In this section, the PLC functions have been divided into fundamental and complex sets. The fundamental functions are those that allow the PLC to replace the traditional industrial

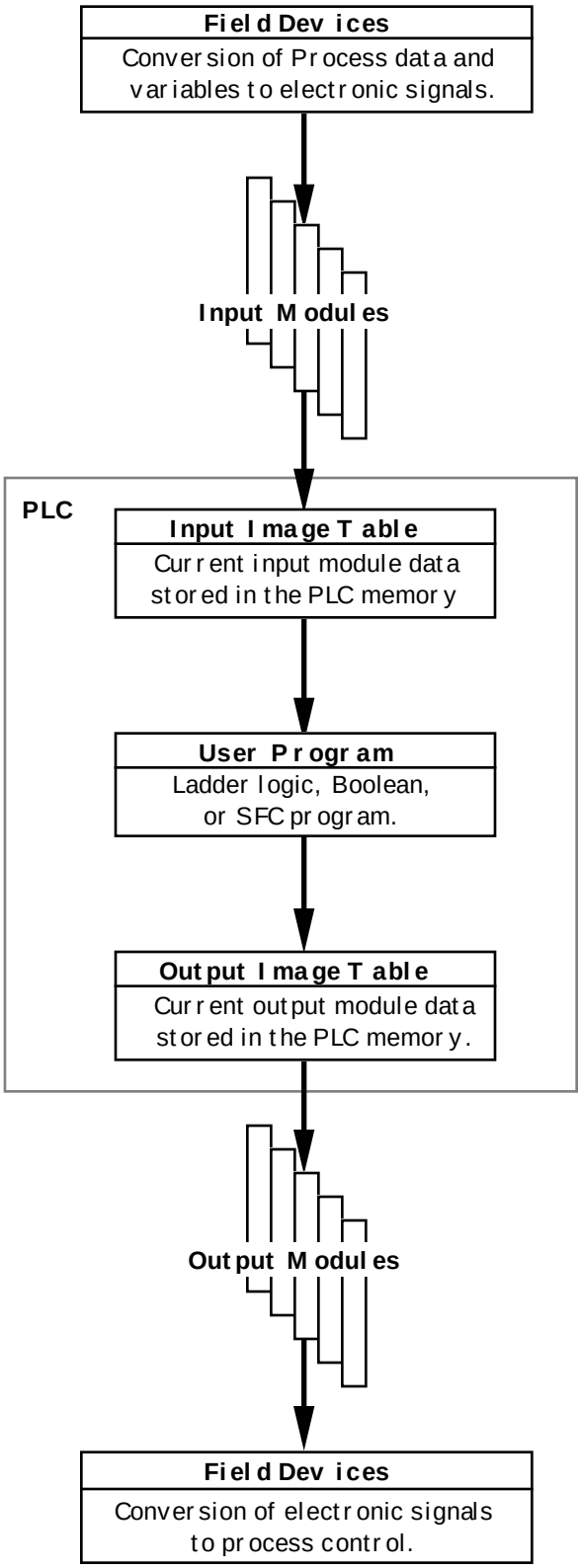


Figure A-1. PLC Run-Time Operation.

control relay applications. The complex functions have evolved from more sophisticated control applications over the past years.

3.4.1. Fundamental Functions

The functions listed below are basic to all PLC systems. These functions represent the original intent of the PLC, which was to replace industrial control relay circuits.

Standard Relay

The standard relay is very simple in construction and its function is easily implemented on a PLC. The significant components of all relays are the coils and their contacts. The coil is energized and deenergized, opening and closing the relay contacts. The contacts are commonly called “normally open” or “normally closed,” depending on their electrical operation. The normal position is the state of the contact when the coil is deenergized, also referred to as the shelf state (a term taken from the perspective of seeing the relay in a store on a shelf). The relay has no power and the relay contact position at this time is called the normal position.

Industrial control relay manufacturers refer to relay contact configurations as Form “A,” Form “B,” or Form “C.” Form “A” and Form “B” are normally open and normally closed, respectively. Form “C” contains normally open and normally closed contacts in a single-pole-double-throw configuration. A Form “C” relay has three connection points, one for the pole, one to make a normally open contact and one to make a normally closed contact. Form “C” is not supplied as a PLC function, but is easily implemented using a normally open and normally closed contact referenced to one coil.

The standard relay performs logic and isolates electrical circuits from each other. A control system implemented with relays has relays that are controlled by field input devices, relays that control field output devices, and relays that merely perform logic. An equivalent system implemented with a PLC would use the I/O modules to provide isolation and interface to the field input and output devices. The relays used purely to perform logic would exist in software only. Thus, many of the relays that would be incorporated in a relay-based control circuit are physically eliminated, existing only within the PLC software. The PLC software uses reference designations on the coils and contacts to associate contacts with their respective coils.

The user programs relay coils and contacts into the PLC as needed by the control scheme. The relay coils are controlled by other relay contacts or input module contacts, which in turn control more relay contacts or module outputs. It is important to understand that the relay contact states follow the state of the coil. A normally open contact stays closed as long as the coil is energized. As soon as the coil is deenergized the contact opens.

Latch

The latch relay holds its state until it is unlatched. Typically, a latch relay has two coils, one that “latches” the relay and one that unlatches it. Once the latch coil is latched, it may be energized and deenergized any number of times without affecting the state of the contacts. It takes subsequent unlatching to change the states of the contacts. For example, a normally open contact is closed when latched and will remain closed until the relay is unlatched.

Typically, the PLC contains two instructions to implement the latch function, one instruction to latch and another to unlatch. The latch, unlatch and associated contact instructions have a reference designation to indicate that they all act as one relay.

One-Shot

The one-shot relay changes the contact state for one scan cycle. Typically, two types of one-shots are provided, leading-edge transition and trailing-edge transition. A transition from deenergized to energized is considered to be a leading-edge transition, while going from energized to deenergized is a trailing-edge transition. When the input coil receives the correct transition, the contact transfers state for the remainder of the scan cycle. At the end of the scan cycle, the one-shot resets to its original state.

Timer/Counter

Two basic timer functions exist, “on-delay” and “off-delay.” The on-delay timer has timed output contacts that hold their states for a specified delay time after the timer is *activated*, while the off-delay timer contacts hold their states for a specified delay time after the timer is *deactivated*. Figures 2a and 2b show the timing diagrams for their respective timers. Both types of

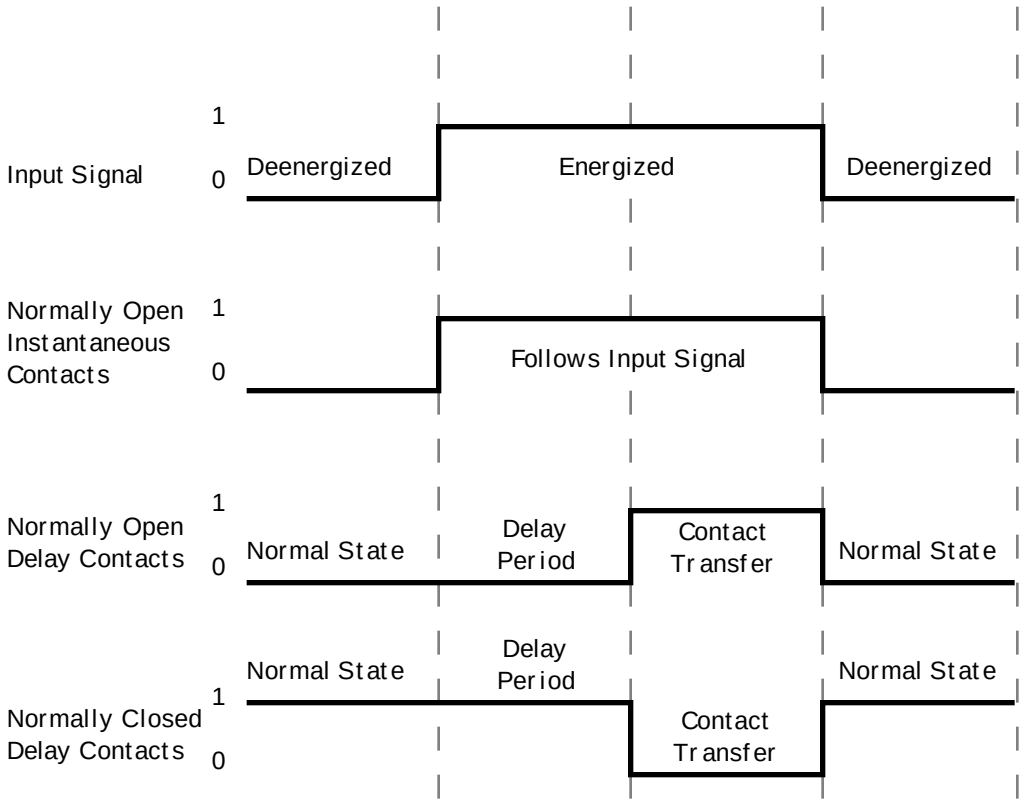


Figure A-2a. On-Delay Timer Operation.

timers have an input signal, instantaneous contacts, and delay contacts. The contacts may be normally open or normally closed for most PLC systems. The instantaneous contacts change state immediately upon energization/deenergization of the timer. For a normally open contact, if the input signal is energized, the contact is closed. The instantaneous contacts may not be provided by the manufacturer, but the contacts are easily implemented with a standard relay and one of its contacts. The delay contacts depend on the type of timer. The on-delay timer delays operation of the contacts for a specified time after energization, and for the off-delay timer, the delay comes after the timer is deenergized. When the on-delay timer is energized, the instantaneous contacts change states, and the internal timer starts. After the elapsed delay time, the delay contacts transfer. The delay contacts hold until the timer is deenergized.

Counters simply count up or down. Energization of the counter input initiates counting. Each counter has normally open or normally closed contacts associated with it. The contact transfer occurs after completion of the count, which may be up to a preset value or down to zero. Depending on the manufacturer, counters are provided to count scans or time. The current market supports time counters with resolutions as high as 0.001 seconds. When using resolutions this high, careful attention must be paid to scan times. For instance, if the scan time is on the order of 0.1 seconds and control of an output is desired on the order of 0.001 seconds, a problem exists—the controlled output will never respond in time because of the inherent time-limiting nature of the PLC scan cycle.

Retentive timers are common. These timers react once to their input condition, then they must be reset in order to react again. Both timers and counters are available as retentive functions. For timers, the timed contacts respond once after the initial delay and then hold that state until a reset signal occurs. Figure A-3 shows a timing diagram for a retentive timer.

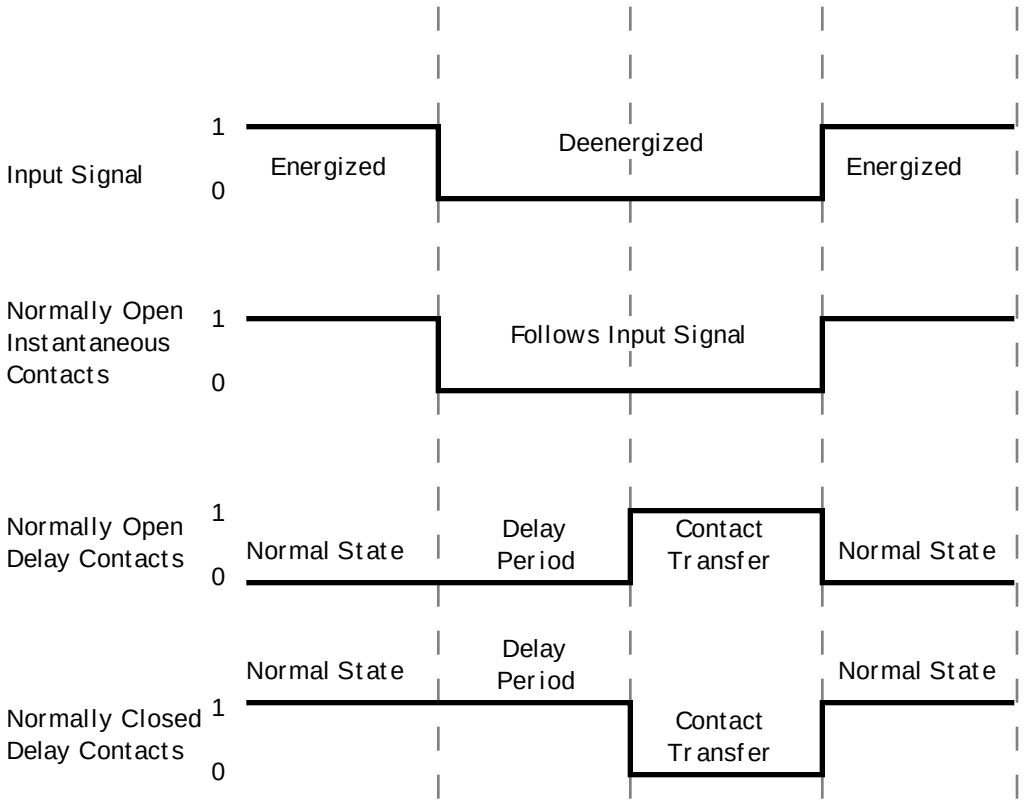


Figure A-2b. Off-Delay Timer Operation.

Some key operational concerns about timers and counters are worth noting. The reset condition overrides the time or control input. As long as the reset is active the timer or counter will be in the reset condition, typically the shelf state for the output contacts. In addition, caution must be taken when restoring back-up copies of software that contain timers or counters. Some PLCs store the current timer/counter values when backing up, and restore that value, while others do not store current values and instead restore reset values. The user must make sure the restored values are the desired ones. One other key operational note is that when the PLC is in program mode, most PLC systems automatically reset the timers and counters, which may not be desirable.

3.4.2. Complex Functions

Data Manipulation

Move functions transfer data between areas of memory. These areas include, PLC RAM, I/O module RAM, hard-disk file, PLC CPU register, and I/O CPU register. Each function transfers one element or group of elements between any two memory areas. Some instruction examples are:

MOVE	Moves a single element from one memory location to another.
BLKMOVE	Moves a group of elements of the same format from one memory location to another.
TBLMOVE	Moves one element of a group of elements to a single element memory location.
SWAP	Swaps two single elements.

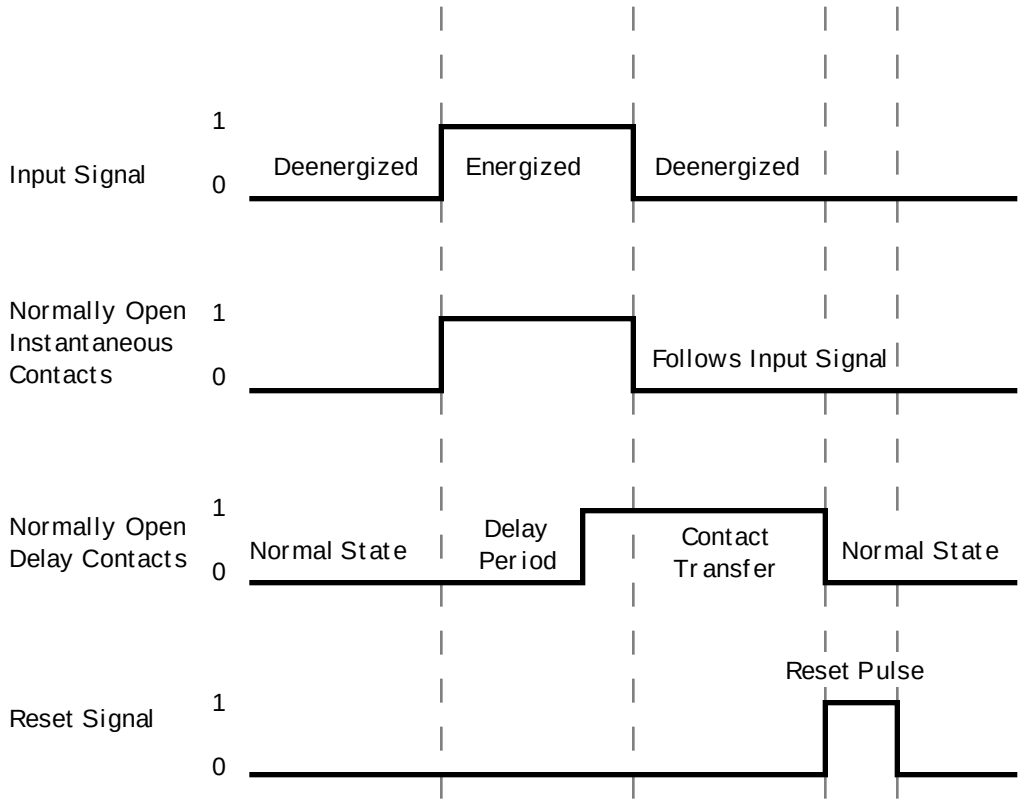


Figure A-3. On-Delay Retentive Timer Operation.

Shift functions manipulate data contained in tabular form. Tabular form data is stored in a block of consecutive memory locations. The data is shifted up or down. The input and output to a shift function are designated as source and destination, respectively. The source and destination are typically any memory location or I/O point. Shift-up outputs data from the highest memory location to the destination, shifts all data to the next highest memory location, and inputs data from the source to the lowest memory location. Shift-down is the reverse. Data from the lowest memory location is shifted to the destination, all data is shifted to the next lowest memory location, and source data is input at the highest memory location. The shift direction, destination location, source location, table starting location, and table size are usually provided by the programmer.

Search functions are exactly that. They search for specific data over a specified range of memory locations. The range could be specified as a file on hard-disk or a table in PLC RAM.

Queue functions are similar to shift functions, except they operate on a particular area of CPU memory called the queue or stack. The common queue types are FIFO and LIFO, “first in, first out” and “last in, first out,” respectively. LIFO queues are sometimes called “push down stacks.” The queue has a maximum capacity. Associated with the queue are two instructions: load and unload. In FIFO, a load instruction loads one data element on to the top of the queue and unloads one data element from the bottom. For LIFO, the top element of the queue is loaded and unloaded. Most queues have indicators that monitor when the queue is empty or full. A conscious effort must be made not to overflow the queue on PLC systems with no queue full indicators. PLCs with no queue full indicators are not recommended for use in safety shutdown applications. It is the programmer’s

responsibility to test for the queue overflow, as well as other error conditions, and take desirable actions.

Bit Manipulation

Manipulation and comparison of individual bits are some of the most useful functions in a PLC system. Bit operations may be performed on data words (typically 16 bits) or tables of data words. Bit operations can be very time-consuming. In order to keep the scan time to a minimum, the bit operations are divided into modes of execution. Three popular modes are incremental, continuous, and distributed. Incremental mode operates on one data word at a time. Continuous mode processes an entire table in one scan. Distributed mode breaks up the table into smaller sets and processes the data over more than one scan. The list below specifies the most common functions:

- ORing
- Shift Left/Right
- ANDing
- Rotate Left/Right
- EXCLUSIVE ORing
- Comparison
- COMPLEMENTing.

ORing, ANDing, EXCLUSIVE ORing, and COMPLEMENTing are standard Boolean logic operations performed on two words or two tables bit by bit. Shift and rotate are standard computer shifts on data words or tables. The difference between shift and rotate lies in the control of the end bits. For shift, the end bits are input by the CPU and output to the CPU. For rotate, the end bits cycle around to each other.

Comparison examines any two words or two tables for a bit mismatch. Two status contacts are typically provided. One status contact indicates that a comparison is in progress and the other indicates a mismatch occurred. When a mismatch is found, the comparison function specifies the position of the mismatch. The position of the mismatch is usually logged into an error file and/or displayed on an operator's screen. Comparison allows diagnostics of the field devices. The I/O modules communicate to the PLC CPU with 8- or 16-bit words. For discrete I/O, each bit of the word represents an I/O point. At various stages of the process (especially at start-up), these discrete I/O words or tables may be compared to predetermined words or tables to verify proper operation of the field devices. If a mismatch is detected the PLC can correct for the problem or shut down the process.

Math Functions

The four basic math functions of addition, subtraction, multiplication, and division are usually supplied. Additional functions may be modulo-division, remainder, square root, and negate. Also furnished are the standard equality operators of equal to, greater than, less than, greater than or equal to, less than or equal to, and not equal to. Typically, math functions require specific registers to be used for operands and results. This requirement, along with the mechanics of setting up the math functions, are typically cumbersome.

Type Conversion

Most of the time, the PLC data is displayed in decimal format, but the PLC may store the data in a number of different formats. Any computer data format may be used. Some examples are BCD, ASCII, unsigned decimal, integer, floating-point, double, etc. Due to this, most PLCs provide type conversion functions that convert the data from one format to another.

Control Functions

"Control functions" covers an ever-evolving area of control algorithms. Common in virtually every PLC is the popular PID process control algorithm and its associated variations of ratio, cascade, adaptive, and feed-forward control. Another popular control function is the sequencer, which controls a set of digital outputs by the use of data tables. Each column of the data table represents the states for one digital output and each row of the data table is one sequence. Via the user program, the sequencer progressively steps through each row of the table at various scan times. Thus, the digital outputs are sequenced on and off. The program dictates the time interval between steps of the sequence, while the sequencer contains the control information for each sequence.

Diverse control functions exist from various manufacturers. Implementation of control functions may be in a neatly packaged I/O module, or a PLC software package using analog I/O modules. This functionality is specific to user needs. Wherever a large customer base for a specific control application exists, the PLC manufacturers are sure to respond.

Some PLC control functions provided by various manufacturers are listed below:

- Servo Positioning (with encoder)
- Plastic Mold Injection Control
- Stepper Positioning
- Real Time Clock
- Linear Positioning
- Sheet Metal Press Clutch/Brake Control
- Axis Positioning.

Subroutines

Standard computer subroutine capabilities are offered on most of the larger PLCs. The instructions include jump-to-subroutine and jump-to-label. Jump-to-subroutine allows repetitive calling of subroutines in one scan. Excessive scan times may occur when using subroutines, especially when using repetitive calls to subroutines. Jump-to-label jumps to another location in the program for that particular scan.

Special Languages

In an effort to insert more versatility into the PLC system, various manufacturers are offering BASIC programming modules. These modules allow users to program their particular applications using the manufacturer's proprietary form of BASIC. The BASIC module runs BASIC programs independently of the PLC processor. The necessary input data and resultant output data is the only information passed between the PLC processor and the BASIC module. The PLC processor scans the BASIC module during normal input and output scan times.

Another approach to expanding versatility is providing the PLC system with communications to large mainframe computers. One manufacturer has built a PLC system to the VME bus standard. This system uses the VME standard as the backplane for all PLCs and I/O modules; thus any other VME module may interface to the PLC system. VME is an open-architecture, industry standard, modular instrumentation system. Various manufacturers produce instrumentation and computer modules compatible to VME. VME module types range from DEC VAXstation to IBM PCs to discrete input modules.

When common programming languages such as BASIC, Pascal, or C are used in the PLC system, the software must be developed and maintained under the requirements of any other microprocessor-based system used in emergency shutdown applications. A rigorous development, review, testing, and maintenance scheme such as that outlined in IEC-880 is required and must be followed.

Report Generation

Most PLC systems produce helpful reports. The ladder logic program, Boolean program, or sequential function chart may be printed as seen on the programming terminal screen. Cross references between coils and their associated contacts may be generated. This is very handy for troubleshooting and check-out. Annotation may be added to programs as desired. Other possible reports are I/O module status, program directory, memory mapping locations, and memory usage.

Peripherals

While one manufacturer may provide only a few application-specific peripheral devices, the PLC industry as a whole has an abundant number of these devices. They allow communications to other computer systems, provide a means of program and data storage, furnish hard-copy outputs, aid in debugging and troubleshooting, and improve the users' and operators' environment.

Most manufacturers offer I/O modules or internal PLC hardware that links the PLC system to PCs, large mainframe computers, printers, other PLCs, special-purpose interface boxes, networks, etc. A short list of communication protocols offered in PLCs is:

- RS-232 serial link
- GPIB-488
- RS-422 serial link
- RS-485 multidrop
- RS-423 serial link
- MAP 802.4
- RS-449
- Ethernet
- Manufacturer's Proprietary Networks.

This versatility in communications opens up a wide array of functionality and complexity for a PLC system. Tape recorders, floppy disks, and high-density computer-grade tape cartridge recorders may connect

to PLC systems. The entire selection of printer types, from dot matrix to graphic printers, are available. Simulation panels, simulation modules, and even computer-based simulators are provided to check out PLC systems. Another very popular peripheral is the color-graphic terminal. Color graphics enhance the operator's environment by color-coding the process functionality and giving graphical representation to the tedious lines of programming code. The user's environment has also been aided with portability. Both the programmers and maintenance personnel have access to hand-held or portable terminals to assist in trouble-shooting, calibrating, configuring, and programming the PLC system.

Self-Diagnostics

As the electronic industry improves upon the size and speed of microprocessors, the PLC industry has added more and more functionality to all parts of the PLC system. Self-diagnostics of the PLC hardware is a critical area that has seen much improvement over the past years. PLC manufacturers have added diagnostics of the hardware into the PLC hardware, the PLC software, and the I/O modules. These diagnostic packages verify such things as communications, runtime status, process status, PLC status/hardware/memory, and I/O module status/hardware/memory. Included in the diagnostic packages are reporting functions and fault logging. Some systems take corrective action when a fault is detected.

4. INPUT STRUCTURE

This section describes the structure of the standard discrete and analog input modules. The modules not addressed are the specialized communication modules, such as RS-232 communication modules, master–slave network modules, or any other special module not directly connected to a field device.

Input modules are either discrete or analog. Typical discrete inputs are contacts, limit switches, pushbuttons, and pressure/flow/temperature switches. Analog inputs cover a wide range of applications and functionality. Some typical devices are pressure/flow/temperature transducers, motor control signals, vibration transducer, strain gage, load cell, and various other transducers producing electrical outputs. The input module receives signals from the field input devices, conditions those signals, and isolates them from the PLC processor. The standard PLC input structure consists of six basic blocks, as shown in Figure A-4. Each will be described in detail.

4.1. Field Input Devices

The field input devices act as the "eyes and ears" of the PLC system. They provide the conversion from physical processes to control signals. For a PLC system, the control signals are electrical. Electrical signals appear in many voltage levels and signal waveforms.

4.2. I/O Module Terminations

I/O module terminations provide the interconnection between the field input devices and the PLC system. These terminations come in many designs and configurations. Some are fixed to the module, some can be quickly disconnected, and others are fixed to the support structure of the I/O module. By allowing the terminations to be quickly disconnected or fixing them to the support structure, the I/O modules can be removed and replaced without rewiring. This positive attribute is highly recommended for a PLC system used in emergency shutdown systems. All terminations are typically locked down in some way, such as screw termination blocks. In addition, the wire ends may be terminated with lugs to ensure a positive tight contact that will not loosen over time.

4.3. Signal Conditioning

Signal conditioning varies greatly depending on the manufacturer, the system architecture, and the type of signal conversion required. Discrete input conversion is relatively simple, while analog input conversions are more complex. Some common types of conditioning circuits are rectifiers, resistors, resistor/capacitor/inductor networks, and analog conversion. Rectifiers convert incoming AC signals to the desired processor levels. Resistor networks provide DC level attenuation. Resistor/capacitor/inductor networks remove unwanted noise spikes and reduce false input triggering due to field device contact bounce. Analog conversion comes in two forms, counters or A/D converters. Counters transform pulse train waveforms into a binary number representing the number of pulses per unit time. A/D converters are used on varying DC level signals. They convert the DC level to a binary representation of the level.

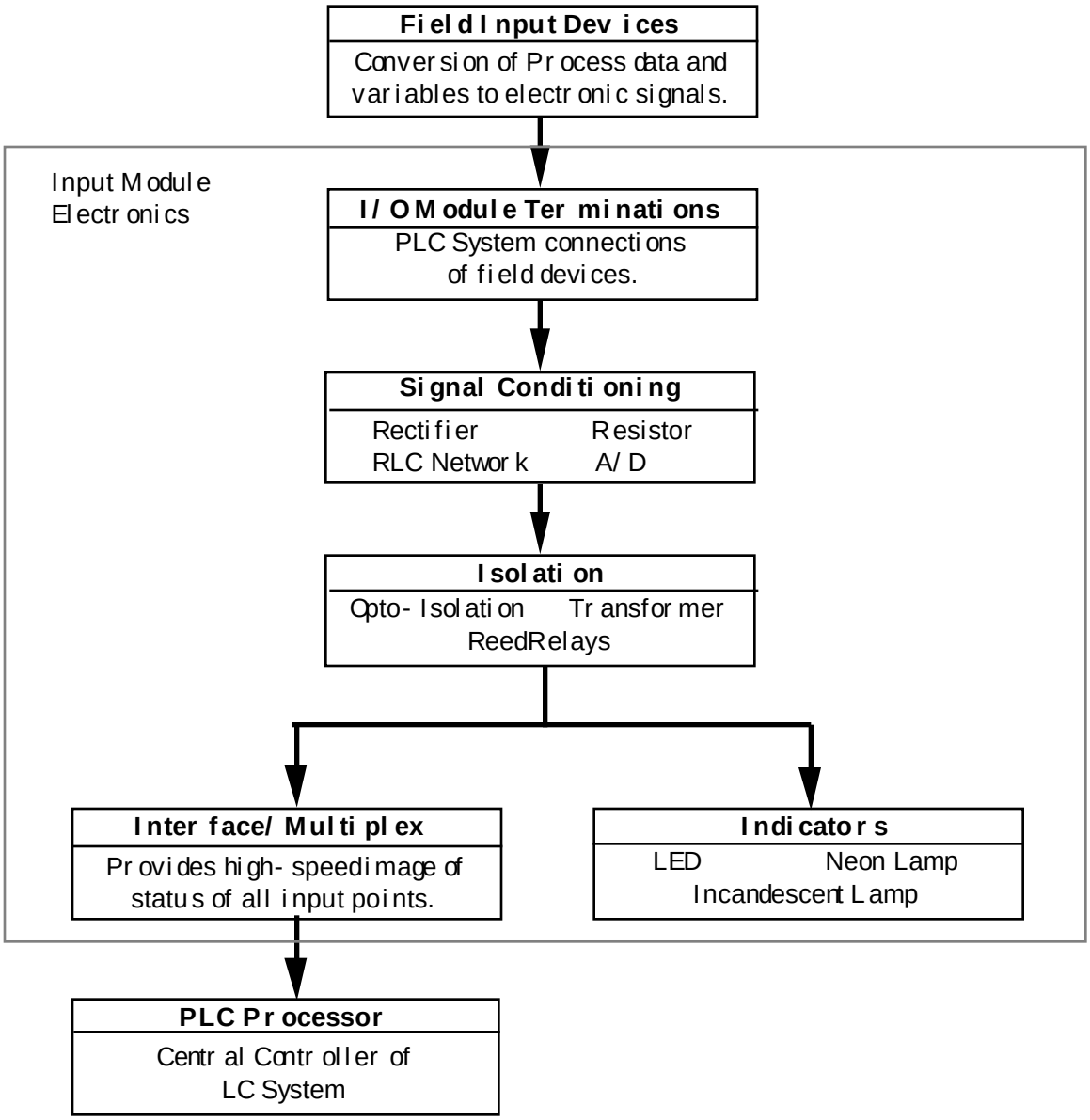


Figure A-4. PLC Input Structure.

4.4. Isolation

Isolation circuits mechanically and electrically isolate the field device signals from the PLC processor signals. Isolation limits the possibility of noise and voltage spikes damaging the sensitive processor electronics. Three isolation techniques are employed, the most common of which is opto-isolation. Before the advent of opto-isolation, transformers were the most common device for isolation, and are still used in a limited number of designs. Reed relays provide a third technique for isolation. A major drawback to reed relays is the limited mechanical life. In the best reed relays the contacts wear out within tens of thousands of cycles. Thus, reed relays are not recommended in intense cyclic operations.

For opto-electronic devices a common isolation test measures the isolation surge voltage, V_{iso}. V_{iso} is a measure of the internal dielectric breakdown rating of the opto-electronic device, not necessarily the I/O module. This voltage is placed across the device as depicted in Figure A-5 below. A typical semiconductor manufacturer's published V_{iso} for opto-electric isolation is about 1500 Vdc for 60 seconds and 1500 Vac, 47 to 70 Hz, for 60 seconds.

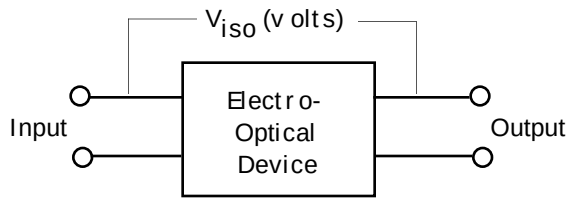


Figure A-5. V_{iso} Test.

4.5. PLC Interface/Multiplex Electronics

This block gathers all incoming conditioned field device signals, and transmits them to the processor as requested. Control signals are produced by this unit and the PLC processor to coordinate the transfer of data. These signals may include clock, reset, enable, module address, handshaking, error-handling signals or data. Some manufacturers provide watchdog timers in this part of the I/O module. The watchdog timer must be reset by the PLC processor. Thus, if the processor does not communicate to the I/O within a specified time period, the watchdog timer expires and the I/O module de-energizes all outputs.

4.6. Indicators

Indicators assist the user in troubleshooting the system and aid in verifying the integrity of the field wiring, the module operation, and the module status. LEDs, neon lamps, and incandescent lamps are all used as indicators. The indicator location varies by manufacturer and I/O module type. It may be located and powered on the field device side, the PLC logic side, or both. The field device side of an I/O module is composed of all the electronics from the termination point to the input of the isolation electronics. The PLC logic side contains all the electronics from the output of the isolation electronics to the I/O module to PLC interface electronics. The best configuration has indicators on both sides of the electronics. As a minimum, an indicator should be provided on the field device side.

5. OUTPUT STRUCTURE

Section 5 describes the standard discrete and analog output modules. The modules not addressed are specialized communication modules (RS-232 communication modules, master–slave network modules, etc.), co-processor modules, or any other special-function module not directly connected to a process control element.

The format of the discussion below is analogous to that of Section 4, Input Structure. Output modules are either discrete or analog. Typically, discrete outputs control relay coils, valve solenoids, motor starters, panel indicators, and alarms. Analog outputs cover a wide range of applications and functionality. Some typical devices controlled are pressure/flow/temperature valves, motor control signals, analog meter and various other applications needing electrical signals. The output module receives signals from the PLC processor, converts and isolates those signals, and controls the field output devices. The standard PLC output structure consist of seven basic blocks as shown in Figure A-6. Each will be described in detail.

5.1. PLC Interface/Multiplex Electronics

The PLC interface/multiplex electronics gathers the PLC processor signals, decodes the address, and passes them to the appropriate output destination point. Many control signals must be provided by the PLC processor to enable this block to function correctly. The signals needed vary depending on the manufacturer’s hardware/software design, but some typical signals are clock pulses, reset, enable signals, addressing data, and error handling data. Also, the interface/multiplex electronics sends reply and status data back to the PLC processor.

5.2. Signal Latching

The signal latching circuitry receives data from the interface/multiplex electronics. This circuitry contains electronic latches such as flip-flops to hold the latest data received. The data is held until the next update of output data. The PLC processor ensures that the latch block is updated within a specified time period.

5.3. Isolation

The isolation circuitry for output modules is identical in design to the input module isolation circuits.

5.4. Signal Conversion

The signal conversion circuitry converts the latched signals to the proper state acceptable to the field output device. Remember, the signal conversion circuitry is on the field side of the isolation circuitry. Thus, the

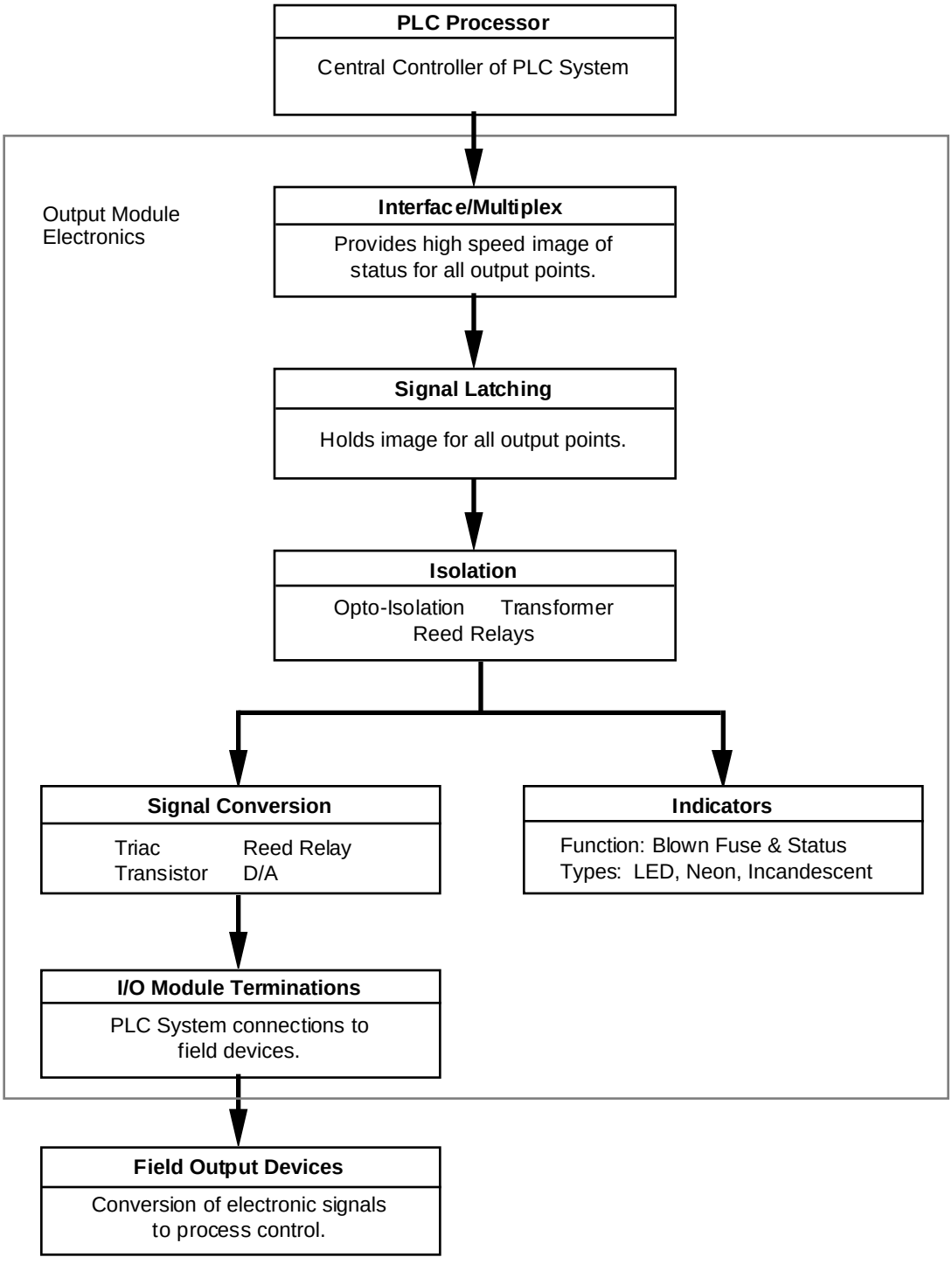


Figure A-6. PLC Output Structure.

power for the signal conversion circuitry must come from the field side. This assures isolation of the sensitive PLC and output module electronics from noisy field devices. The signal conversion circuitry contains a power switch, typically in one of four forms: triac, reed relay, transistor, or digital-to-analog converters. It is a good idea to have a circuit protector connected in series with one of the power switches. The best designs have circuit protectors, such as fuses, in series with each output point.

5.5. I/O Module Terminations

The output module terminations are identical to the input module terminations. Typically, the terminations are locked down in some way, such as screw termination blocks. In addition, the wire ends may be terminated with lugs to assure a positive, tight contact that will not loosen over time. Another positive attribute is the ability to remove and replace the I/O module without rewiring. Many manufactures are making this capability possible by fixing the module terminations to the rack instead of the I/O module.

5.6. Field Output Devices

The field output devices provide the muscle to control the process, and the visual and audio information that expresses the process state. These devices convert the PLC control signals into process changes or status. Process changes are accomplished through control of valves, motors, relays, etc. Process status is presented through graphical displays, horns, lights, and other devices.

5.7. Indicators

As in the input module circuits, indicators are provided to aid the user in troubleshooting. The output module indicators may be configured in the same manner as the input modules; indicator powered from field side of isolation circuitry; indicator powered from PLC processor side of isolation circuitry; or both. The best configuration would have indicators on both sides of the electronics. Some output modules have an additional indicator to detect a blown fuse. The fuse is located on the field side of the isolation circuitry and the indicator connects in parallel to the fuse, so when the fuse blows the indicator lights up.

6. POWER SUPPLY

The power supplies considered are all sources of power necessary to properly operate the PLC system. The first and largest is the signal conversion electronics required to convert 120 Vac or 240 Vac into the low-voltage DC power necessary for operation of the processor and I/O modules.

The power supplies may be located on each module (processor or I/O), contained in the mounting rack, or the processor module may supply power to the I/O modules. The most common configuration of the power supply is not the most reliable, but definitely the most economical. Most manufacturers design the mounting rack such that each rack has one power supply, making this one power supply a single-point failure. If the power supply malfunctions on a one-rack PLC configuration, it takes the whole PLC system down.

Both linear and switching power supplies are used in PLC power supply designs. Linear power supplies use transformers, rectifiers, and various filtering and detection circuits to transform high-voltage AC power into low-level DC power. The switching power supply is a newer design that is physically smaller and has a higher signal conversion efficiency. However, the switching power supply responds more slowly to electrical transients and intrinsically produces more noise, which shows up as low-voltage ripple on the DC output lines and in EMI.

Some important features that should be designed into both types of power supplies are input filters, output filters, short circuit and overload protection, over-voltage and reverse-voltage protection, and incoming line monitoring. Input filters reduce incoming electrical transients, while output filters stabilize the power supply’s low-voltage DC output and reduce unwanted noise. Short-circuit-overload circuits protect the power supply from destroying itself during abnormal current conditions. Likewise, over-voltage and reverse-voltage protection circuits protect against abnormal voltage conditions. Incoming line monitoring is a highly valuable circuit, but not as common as the protection circuits previously mentioned. By monitoring the incoming line voltage and frequency, the power supply can detect power outages and give warning to the PLC processor, allowing the processor to shut down the system in an orderly fashion. During restoration of power, the PLC system can then start up and continue normal operations.

Another common power source for a PLC system is batteries. Batteries are used to back up volatile RAM memory or the real-time clock system. Manufacturers use primary and secondary battery systems. Primary

batteries cannot be charged, while secondary batteries are rechargeable. The PLC system should be capable of producing an alarm for low-battery status locally and remotely to users of the system. It is important to note that remote alarm capability is essential. Often, the PLC system is locked behind cabinet doors and the users rarely see the local displays. A low-battery status at a remote terminal can reduce the probability of one inexpensive battery shutting down the entire plant.

Primary battery types include carbon–zinc, alkaline, and lithium. All need replacement about once a year. The carbon–zinc family is rarely used in current PLC design because alkaline and lithium batteries offer significant advantages. Alkaline batteries offer a lower cost and extended life over carbon–zinc types. Lithium is the best family of batteries, offering three to ten times the energy density, and twice the shelf-storage life of other battery types. The disadvantages of lithium are high cost and explosive hazard—lithium reacts violently with as little as 100 parts per million of water. For this reason, lithium batteries are hermetically sealed and must be transported under Department of Transportation regulations (DOT-E 7052).

Lead–acid and nickel–cadmium are secondary battery types. This category of battery requires a recharging circuit, which makes secondary battery circuits more complex than primary battery circuits. Their replacement period is much longer than primary battery types and depends on battery usage. Lead–acid batteries are big and bulky. Most have gelled electrolyte packaged in a plastic case. Lead–acid batteries do not explode. At worst, the battery overheats, melts the plastic case, and corrodes any surrounding metal and/or electronics. Nickel–cadmium batteries provide the same characteristics as the lead–acid type, but in a smaller package. They are available in standard metal-cased AA and D sizes. Nickel–cadmium battery systems are used in the majority of secondary PLC battery systems.

7. COMMUNICATIONS

As PLCs grew, the requirement to communicate with other PLCs and intelligent external devices, such as computers, color graphic terminals, and other microprocessor based devices spawned the application of a vast assortment of communication networks to PLC systems. A comprehensive discussion of the design trade-offs of communication networks is beyond the scope of this paper. Some of the main concepts and components of different communication systems used in PLC systems will be covered below, as well as a few drawbacks and advantages.

A PLC processor-to-processor network is a high-speed data link designed for passing information between the PLC processor and other devices connected to the network. The information may be in the form of data, control signals, addresses, system status, or individual device status. Most PLC systems use communication rates of about 57 kilobaud, which is sufficient to support most of the process control applications. Manufacturers are pushing the upper bound of the communication rate to 2 megabaud. These high communication rates prove beneficial in moving important data to various components of a safety shutdown system applied to power stations, refineries, steel mills, and the like.

Fundamental differences exist between two networks in a PLC system. One network , called the control network, transfers all information required from one PLC processor to all its mates (I/O, other processors, and other types of modules) within a single processor scan. The second network, called the system network, gives up the ability to transfer all data within one processor scan and may not need to meet real-time requirements. Either network may use the components and routines described below.

7.1. System Formats

Two basic system formats are used in PLC control networks, master–slave and peer-to-peer. Most manufacturers use a proprietary protocol on the control network. PLC system and processor-to-processor networks typically conform to available protocol standards and may use master–slave, peer-to-peer, or other system formats.

7.1.1. Master–Slave

In the master–slave arrangement an intelligent device manages all network communications between all other devices on the network, known as slaves. All slaves communicate to the master and only the master; no slave-to-slave communications are possible. For one networked device to communicate with another, it must transmit its message to the master, who in turn transmits the message to the appropriate slave, a method that gives the master device total control over all communications. A major disadvantage is that the communications is totally dependent on the master device. Thus, a failed master device is a single-point failure that takes down all communications. Some

manufactures offer a second back-up master to operate the network in case the primary master fails. The master device may be a computer or a faster/larger PLC processor. A single point of failure still exists in the switching mechanism, which can be composed of hardware or software.

7.1.2. Peer-to-Peer

Each device in a peer-to-peer network has the capability to directly communicate with any other device on the network. Any number of devices may fail and the network should continue to function. As long as two devices remain on the network, communication is possible. This control scheme is harder to implement because issues such as which device controls the network, how long a device may control the network, and the type of information passed on the network must be worked out. This scheme of communications is often referred to as baton or token passing, since control of the network is passed (like a baton passed in athletic track events) from device to device.

7.2. Components

The various components of a communications network are briefly discussed below. Any communication network used with computers can be used with a PLC system. The components mentioned below are the most common used in PLC systems.

7.2.1. Transmission Medium

The transmission medium is the physical conductor used to convey information between various locations. Some common media are twisted shielded pair, coaxial, triaxial, and twinaxial wire/cable. The control network also uses a backplane, which is a short-length transmission medium fabricated by printed circuit board and/or wire-wrapping techniques. In special applications, telephone, radio wave, or even microwave is used as the transmission medium. Recently, manufacturers are supplying fiber-optic communications. Fiber-optic communications offer large bandwidths, immunity to EMI, and complete electrical isolation. Some disadvantages are the high cost and fragile nature of the cables and connectors.

7.2.2. Interface Electronics

Most communications networks contain a network adapter module or modem to access the data from the network and pass it on to the PLC processor. The modem handles handshaking, error checking, media use management, and network protocol.

7.2.3. Configuration

Many configurations are available. The daisy-chain configurations tie the network devices together one by one, so each device has two connecting neighbors, except the end devices, which only have one neighbor. This configuration is similar to a string of Christmas tree lights, where each light represents one network device. By connecting the two end devices of a daisy-chain arrangement a loop configuration is created. Multidrop systems are similar to daisy-chain arrangements except that at every device a signal splitter is used. The splitters allow all devices on the network to receive all messages transmitted on the transmission medium. Another configuration, sparsely applied in the PLC industry, is the Star configuration. One device is a central point to which all other devices are wired, like the spokes in a wheel. Star configurations are easily adapted to master–slave formats. Any of the mentioned configurations may be used for the control network, the system, or processor-to-processor network.

7.3. Error Handling

A communication system without error handling is extremely susceptible to the transferal of erroneous data, which most likely will lead to incorrect control of the process. Error-handling algorithms have been developed to correct for this. In the control and processor-to-processor network, the error-handling routines are designed into the network by the manufacturer. For the system network, the user can usually select the error-handling routines desired. Two levels of error-handling exist, error detection and error correction. The error-handling routines common to PLC systems are discussed below.

All of the popular routines used in computer systems are used in PLC systems. Parity checks, often called vertical redundancy checking (VRC), longitudinal redundancy checking (LRC), and cyclic redundancy checking (CRC) are used for error detection, while forward error correction (FEC) and automatic repeat request (ARQ) are used for error correction.

The control network typically uses CRC with ARQ. CRC is the more complex algorithm and yields the highest error-detection rate of the three algorithms. Typically, ARQ is used with three re-transmissions. The sender will re-transmit upon receiving a negative

acknowledge or not receiving a acknowledge after a specified period of time. Typically, the sender retransmits three times before shutting down the communication link.

The system and processor-to-processor network error-handling routines use any combination of the above error detection and correction routines. A few PLC systems allow the user to program the desired error detection in the system network. The user selects no error detection, VRC, LRC, or CRC, and the PLC system usually adds ARQ error correction. Again, most implementations of the system network use ARQ error correction with three re-transmissions.

One reason ARQ error correction is more common than FEC error correction is the substantial overhead burden of FEC. FEC requires nearly one overhead bit per transmitted bit, which puts an extreme burden on the communication bandwidth.

7.4. I/O Configurations

The I/O configuration plays an important role in design of the communication network and choice of error handling routines. Two configurations are explained below.

7.4.1. Local

Local refers to the physical location of the PLC processor in relation to other modules. In addition to standard I/O modules, the other modules may be co-processors or even other PLC processor modules. Modules installed within 75–100 feet of the PLC processor, determined by cable length, would be considered local configurations.

7.4.2. Distributed

Distributed or remote I/O configurations are PLC systems where two or more PLC systems are linked together to form a larger system. The total cable length for some distributed PLC systems can be as long as 20,000 feet. This is extremely long for a cable run, but longer lengths can be obtained through telephone lines, radio waves, or microwaves. Another distinguishing feature of the distributed system is that some sort of system communications exist to allow data transfer between PLC systems. Typically, a specific module in each PLC system directs all the system communications, including transmission to and from PLC systems, error handling, additional communications overhead, and communications with its superior local PLC processor. This frees the local PLC processor to manage its local I/O modules.

7.5. Capacity

The amount of data that a master–slave or peer-to-peer format can transmit depends on the design of the system. A communication system transmits messages that consist of data and overhead. Overhead refers to the bits required to accomplish such things as error detection, error correction, and message decoding. Data is the remainder of the message; the actual data the receiver needs to accomplish its task. Data is typically transmitted in blocks of words. A word consist of 8, 16, or 32 bits, and a block contains many words.

The data capacity of a communication system depends on such constraints as modem design, communication protocol, error-handling algorithms, and processor data-handling capabilities. For this reason, the rate of data words (words to be used by the receiver of the message stripping away all the overhead required to send the signal) transmitted, not bandwidth, is a most important specification of the communication network.

8. PERIPHERAL DEVICES

Peripherals aid in programming, storing data, printing, emulation, simulation, or other functions the user may need or want. In addition to PLC manufacturers, a number of third-party vendors develop peripherals.

8.1. Programming Terminals

The single most important peripheral is the programming terminal. The programming terminal accepts the program commands of the user, converts them into machine-readable code for the PLC processor, and allocates the machine code into the appropriate memory locations within the PLC processor.

The most common programming terminal in the PLC industry is a form of the video display terminal (VDT). Most PLC programs are developed on VDTs. Recently, smaller programming devices have been developed. This new group of programming terminals are hand-held and use liquid crystal displays. The hand-held terminals compact size makes them very useful in the field.

The PROM programmer loads PLC programs into PROM devices. After programming the user inserts the

PROM device into the appropriate PLC circuit board socket. PROMs eliminate reprogramming by the casual user. Once the program is locked into PROM, a new PROM device must be physically inserted to change the user's program. This level of program control and safety may be desirable in safety shutdown applications.

8.1.1. Modes of Operation

As in all computer systems, various modes of operation exist in the programming terminal. The two most common modes are on-line and off-line. On-line and off-line refer to the mode into which the PLC processor is placed by the programming terminal. Other modes of operation may be supervisory, executive, or monitor. These modes limit the user's access to the PLC system. For example, supervisory mode may only allow configuration of the I/O modules, and monitor mode may only allow viewing of the program flow and I/O states.

On-line programming allows the user to add, change, or delete the PLC program or data while the PLC is in run-time. The user must be very careful not to cause undesirable actions in the control process. Most terminals let the user monitor the new program changes before downloading them to the PLC processor. This way the user can monitor the changes until he or she is sure the correct process control will result. In safety-critical applications, tight control must be maintained on all devices that allow on-line programming.

No on-line programming changes should be made after start-up unless the on-line mode is deemed absolutely necessary, and then only with the appropriate written approvals of the change and use of the process. The personnel making the change must fully understand the process, the equipment being controlled, and the operation of the PLC system. The change should be checked for accuracy and all possible process control scenarios should be exhaustively thought through.

Documenting the change beforehand aids in thinking through the change at the different development levels. The development levels include process control specifications, piping and instrument diagrams, ladder logic drawings, and the ladder logic program listings. By going through the thought process potentially dangerous changes may be caught before actual implementation. Although this is a very precarious mode, it offers great flexibility when troubleshooting the system.

Off-line is the more common programming mode. The user keys in the program to the programming terminal with no effect to the PLC processor. After the program is written the user may download the program into the PLC processor. While in the off-line mode, the programming terminal will cause the PLC processor to stop scanning and de-energize all outputs before downloading.

Once a program is running on a PLC system, care must be taken when switching to off-line mode. When placed off-line, many PLC processors reset the elapsed time of counters and timers. In addition, the outputs return to the deenergized state. If some of the controlled equipment was in an intermediate state, it may malfunction or be damaged when going off-line. Initialization and shutdown routines may be helpful. These routines place the PLC outputs in a predetermined safe state during start-up and before shutdown, respectively.

8.1.2. Select Commands

A number of commands are provided by each manufacturer. Some are necessary to program the PLC while others add user conveniences in each of the stages of development, debugging, and operation. Detailed discussions on every possible command are beyond the scope of this paper, but one command is of particular importance because it may effect real-time control of the output modules.

The "Forcing" commands hold an input or output at a desired state. These commands are typically FORCE ON/FORCE OFF or ENABLE/DISABLE. Typically, the user requests I/O forcing from the programming terminal. Once requested, the programming terminal downloads the forced I/O points into the appropriate PLC I/O image table. Some PLCs keep the I/O point forced for one scan, after which time the I/O point is automatically returned to normal operation. Other PLCs require a FORCE OFF command to return the I/O point to normal operation.

Forcing is useful in debugging the program and verifying proper operation of outputs and their connected equipment, but can be disastrous if not used properly. Forcing after installation causes real process events (valves to close/open, motors to start/stop, etc.) to occur. Important equipment such as safety interlocks, motor hold-in contacts, and other safe circuits can be bypassed or modified by forced conditions, increasing the probability of damage to property or life. Forced conditions should be avoided

as much as possible after the PLC has been installed. In addition, once an I/O point is forced it may not be returned to normal operations. Most PLC systems remove all forced I/O points once the programming terminal is switched off or disconnected from the PLC communications port. The user must be knowledgeable about the plant process, the PLC system, and the characteristics of the forcing command to properly use forcing. Further, in safety application, the forcing commands should be used only with strict administrative control.

8.2. Select Devices

Tape drives, floppy drives, hard-disk drives, and compact disk drives are all available to provide storage of program and PLC system status. Printers of all sorts and varieties are available. Some newer and perhaps more interesting devices are graphic display terminals, emulators and simulators.

Graphical displays are used to monitor and sometimes control the plant process. The display represents motors, pipes, valves, tanks, and other process equipment with graphical objects. Important data is overlaid on those objects. The PLC system network requires high data transmission rates to support graphical displays. If the system network is not fast enough, the terminal may not display the most current data. In addition, the communications between the PLC processor and the graphic display terminal may delay the processor's reaction to a system fault or required process change.

A few PLC manufacturers offer simulators, but most simulators must be purchased through third-party vendors. Simulators may be either special packaged units independent of the PLC system or I/O modules that mount directly into the PLC racks. The packaged units can range from simple knobs, switches and indicators to complex CPU-based simulators. I/O module simulators replace a specific manufacturer's I/O module and simulate the module's operation to the PLC processor. Simulators are used to verify correct operation of the user program, the PLC system (including the processor, communications, and the I/O modules), and the field devices.

Emulators are software-based tools used to verify proper operation of the program. Emulators allow the programming terminal to operate in the on-line mode without being physically connected to a PLC system. The emulator reads and solves the logic program and feeds back the necessary display information to the programming terminal. The PLC processor or its I/O modules are never communicated with nor controlled.

Both simulators and emulators are useful tools in troubleshooting and debugging PLC software and hardware. In addition, simulators and emulators can be used to play "what if" games and validate correct system responses to presented threats.

Personal computers with software to aid in the design, implementation, debugging, emulation, and documentation of the application program are often offered by the PLC manufacturer.

9. PROGRAMMING

All PLC manufacturers offer programming languages, which attempt to couple program commands with industrial control functions. Three languages are commonly used in the PLC industry: Boolean, ladder logic, and sequential function charts. Boolean programming dominants the Japanese markets but is rarely seen in the United States or Europe. Ladder logic is the language of choice for U.S. manufacturers. The most common language in Europe is sequential function charts. Japan and the United States are also tending towards sequential function charts; some U.S. manufacturers offer both ladder logic and sequential function chart programming.

9.1. Boolean

Boolean programming is a lower-level language. It is similar to programming in machine language, but it uses fewer commands. Boolean programming uses common Boolean operators such as AND, OR, NOT, NAND, and NOR. Other arithmetic and logic commands such as ADD, MULTIPLY, DIVIDE, LOAD, JUMP, and COMPARE are provided. An example of Boolean and arithmetic operations in Boolean code is shown below:

LOAD	I loads the status of the binary input I into a special PLC memory location.
AND NOT	J performs a Boolean "AND" operation on the special memory location with a negated binary input J, I AND NOT J.
OUT	L sets the binary output L to the status of the special memory location. L is set to the result of I AND NOT J.
LOAD	A loads the status of A into the PLC processor's accumulator.

ADD B adds the status of B into the PLC processor's accumulator, adds B to A.

STORE X stores the accumulator status at RAM memory location X. The result of B plus A is stored at X.

A user with a computer background may prefer Boolean programming because of its similarity to common source language programming. The difficulty with programming in Boolean is that the plant process control may be difficult to understand. Additional information such as flow charts, English words, or some other descriptive information may be needed.

9.2. Ladder Logic

Ladder logic was developed specifically to give the process control engineer a programming language that closely parallels process control drawings. This way a typical process control engineer will understand the PLC program with very little effort devoted to learning the language. Ladder logic program commands were developed from ladder logic diagrams. An attempt was made to make ladder logic functions a symbolic representation of the equivalent ladder diagram symbols. This symbolic representation appears in simple relay functions where coils and contacts have symbolic representations.

Other, more complex functions such as timing relays and counters are handled a number of different ways depending on the manufacturer. In one case the complex functions are represented by a square block, and in another case they could be represented by a coil with special mnemonics. In either case, the necessary control information is input in or near the ladder logic symbol. Figure A-7 is an example of a ladder logic program.

9.3. Sequential Function Charts

Sequential function charts (SFCs) elevate the PLC programming language to a higher level. A top-down, object-oriented approach is taken to development, integration and documentation of the process control. The user diagrams the process in the proper sequence, using SFC steps and transitions. A step corresponds to a process control task, and a transition corresponds to a condition that must occur before the PLC processor can execute the next step or steps. After outlining the process, the user programs each step and transition in lower-level graphical tools representing PID loops, switches, other PLC functions, or I/O points. Some U.S. manufacturers program the steps and transitions using ladder logic. In this case, SFCs are a top-level diagram showing the ladder logic program flow, just as flow diagrams show computer program flow. Figure A-8 is an example of a generic sequential function chart. Some of the complexities peculiar to this example are not explained here, as each PLC manufacturer implements SFCs in a different way.

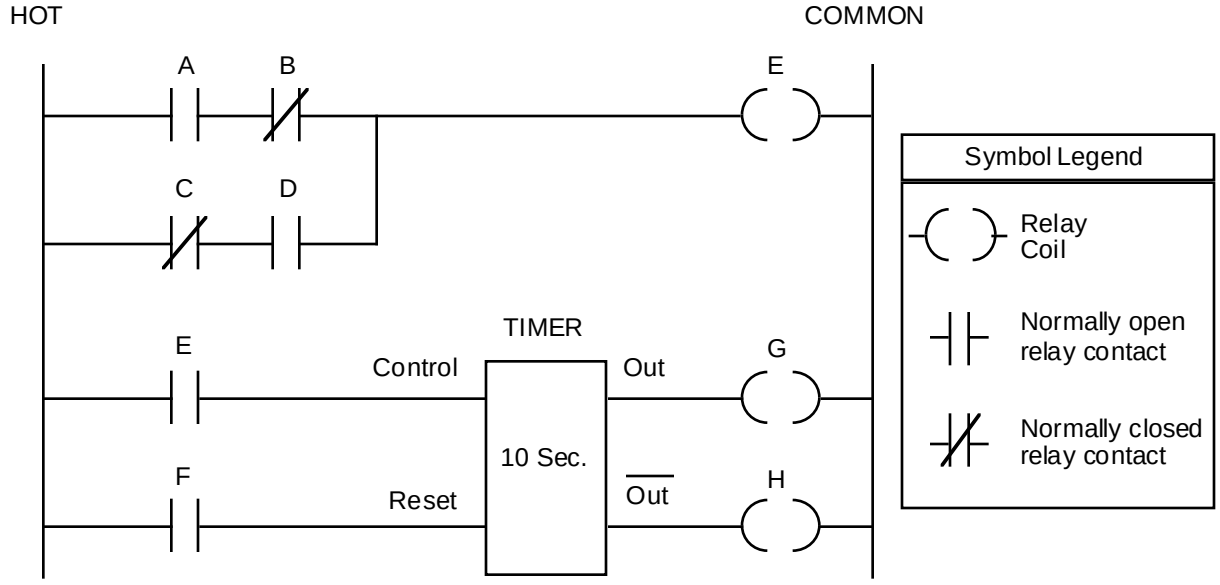


Figure A-7. Ladder Logic Program.

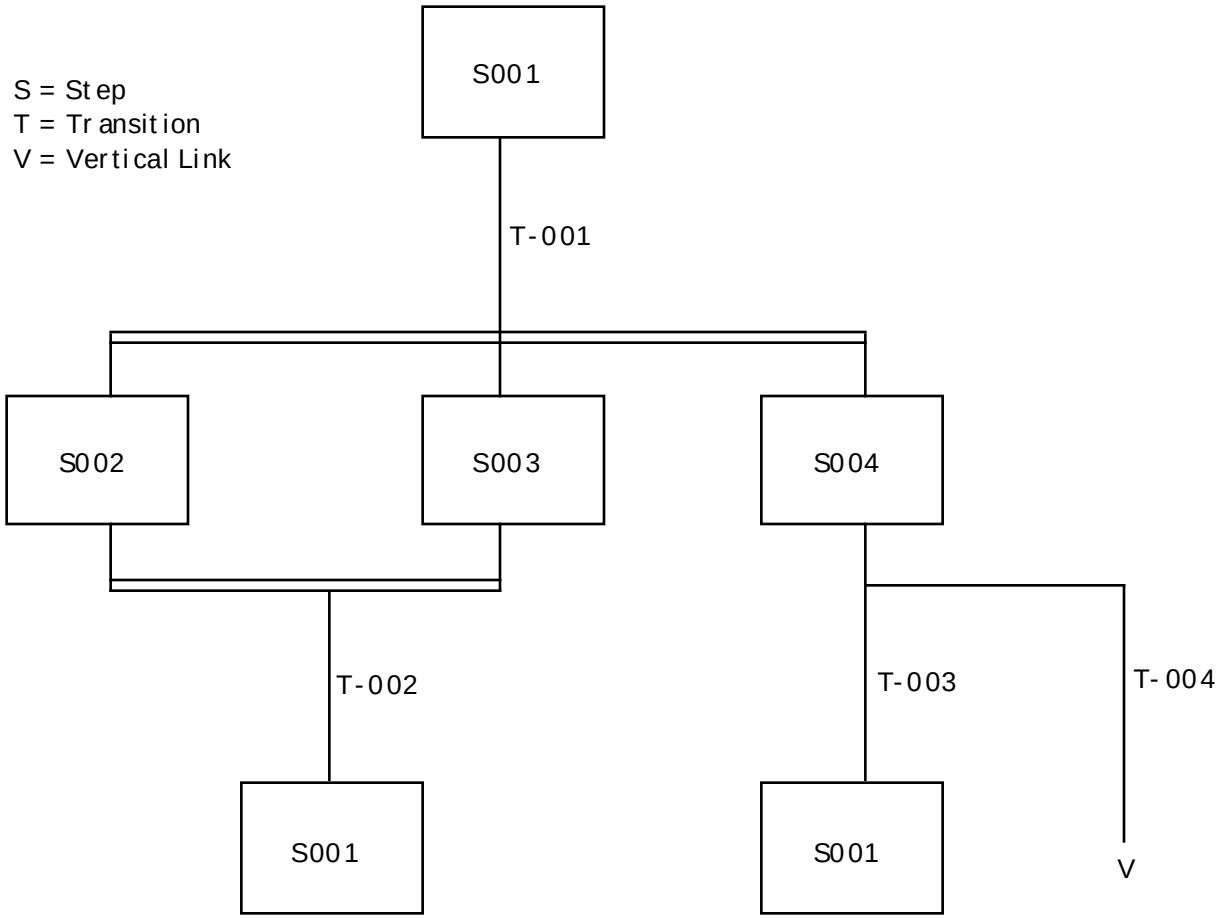


Figure A-8. Example Sequential Function Chart.

9.4. Common Source Languages

Common source languages are now being used in PLCs. This is a trend back toward the computer industry. The language, C, is being incorporated into ladder logic programming. A subroutine block is designated in the program. The subroutine symbolized by the block is written in C. Once the block is programmed, it becomes another PLC function available to the ladder logic programmer. A typical subroutine block will have a controlled input, binary status outputs, analog input data, and analog output data. The controlled input initiates the subroutine. The binary status output indicates such states as subroutine executing, subroutine complete, or subroutine error. The analog input data can be a constant, a memory location, or an analog input point, while the analog output data is sent to a memory location or an analog output point. The C program block allows the user to program any control algorithm into the PLC and allows the programmer to gain access to any part of the PLC RAM, including the memory locations used to support the operating system.

In addition to function blocks, smart I/O modules are available that allow common source language programming. The programming languages used are BASIC, C, or some derivative of these common source languages. These I/O modules are mainly used for control of peripheral devices, but some have been developed to allow quicker real-time response by the PLC system. All of the smart I/O modules communicate with the PLC processor over the control network and are accessed by the PLC processor at least once per scan cycle.

10. SPECIAL FEATURES

Software development tools for the PLC make creation, documentation, testing and modification far easier than it would be for a system implemented with physical relays.

Some special features have been designed into PLC systems to help reduce the probability of error when using, maintaining or accessing the PLC system. Parts of the PLC RAM may be protected and passwords are used to allow access only to password holders.

For example, one manufacturer provides four levels of access on a particular PLC system. This safety measure prevents unauthorized personnel from accessing and/or changing certain programs. Depending on the user's access level, the user is restricted to certain areas of operation which may be: PLC programming, I/O configuration, reading PLC programs, reading I/O configuration, reading PLC data, printing, accessing network devices, and accessing remote I/O.

A key lock on the programming terminal is a simple example of two-level access. The user must have a key to switch the programming terminal into a mode that allows PLC programming. Without the key the user can only monitor the PLC data and program flow.

A special feature to prevent inadvertent I/O module replacement is mechanical keying of the I/O module and the slot. The I/O module and its associated slot in the I/O rack are mechanically configured so no other type of I/O module will physically fit into the slot. Further, most PLC processors contain a program routine often called a Traffic Cop. The Traffic Cop scans the I/O modules and validates the I/O configuration against the current I/O table. If a mismatch is generated, the Traffic Cop shuts down the PLC and reports the problem. Also, the Traffic Cop allows the user to set-up the I/O configuration and specify the interval in which the Traffic Cop will scrutinize the I/O rack configuration.

11. FAILURES

As in other electronic devices, the PLC can achieve a very high degree of reliability, but failures still occur. High reliability along with failure detection can significantly increase the system's availability. This section looks at some of the failures that may have a significant effect on non-redundant PLC systems. These failures, together with possible remedies, may also be used in redundant systems (discussed in Section 12).

11.1. Stalling

Stalling of the PLC processor is perhaps the most critical of failures. This failure may be caused by a hardware failure in the PLC processor or a programming error. Stalling is relatively easy to detect with a "watchdog timer" built into hardware. If the watchdog timer is not reset by the PLC operating system it "times out." Once timed out, the watchdog timer shuts down the PLC system, and if the system is well designed, it will send an alarm to the operators. If designed properly, watchdog timers provide a quick way to detect processor scan failures as well as the most subtle system hiccups. It is important to understand that the watchdog timer hardware design and software routines are transparent to the user, and the details of implementation are seldom published by the manufacturers.

11.2. Instruction Mis-Execution

Another failure is the PLC processor's failure to interpret or execute the user-programmed instruction correctly. This is caused by a failure in the PLC processor, but does not cause the PLC processor to stall. The PLC processor may misinterpret one or more bits in an instruction routine and cause execution of a wrong instruction. Complex techniques have been used to detect subtle failures in the PLC instruction set (Wilhelm 1985), but none of these techniques have proven fruitful.

11.3. PLC Memory

PLC memory failures may be as small as one bit or as large as annihilation of the entire RAM. A simple bit failure can cause an instruction to change in meaning or function, or it can cause the inaccurate process data to be sent to the operator or control algorithm.

PROMs are typically validated via checksums. The PLC processor compares the calculated checksum against the checksum stored in the PROM. If a mismatch occurs, the PLC is shut down and a checksum error is reported. Depending on the manufacturer, the PLC executes the checksum algorithm after start-up, after auto-boot, or at specific time intervals when in run-time mode.

RAM memory usually goes through a non-destructive bit pattern test that operates on all RAM, one memory cell at a time. The PLC processor saves the memory cell contents, writes the bit pattern, reads the bit pattern, restores the contents, and compares the write and read patterns. If a mismatch occurs, the PLC is shut down, the latent memory cells are identified, and a memory error is reported.

One maintenance problem is the volatile RAM back-up battery. If the battery is not properly maintained and the power is lost to the computer, the battery may not hold the memory's contents. Then when power is restored, the PLC will be inoperable until the user program is restored.

11.4. Intermediate Memory

In addition to the PLC processor memory, other memory exists in the PLC system. Separate memory may exist for the I/O image tables, and intelligent I/O modules usually contain a processor as well as memory. The I/O image table memory is typically checked by the PLC processor, which places an extra burden on the PLC processor and will certainly increase the scan time. For I/O modules with memory, an on-board processor typically handles the memory checks.

11.5. I/O Address

Some causes of I/O address failures are encoder failures, decoder failures, communication errors, or an incorrect setting of the address dip switch on the I/O module. Address failures are easily handled with the proper diagnostics. A Traffic Cop is an example of a routine which handles I/O address failures. Another approach is to hard-wire the PLC processor address lines to each I/O module. Thus, communication error detection and correction can be eliminated.

11.6. I/O Module

The I/O module input or output circuitry may fail while the communications back to the processor are working perfectly. For inputs, the damaged hardware may be any component along the input path such as field device power, field device, field wiring, termination points, signal conditioning circuits, or isolation circuits. Similarly, the output failures can be caused by the latching, isolation or signal conversion circuits, termination points, field wiring, field device, or field device power. The field components are mentioned because manufacturers are now producing circuitry that will detect certain failures in the field wiring or field device.

In output modules the triac is the most common electronic device used in the signal conversion circuitry. Manufacturers are now starting to include failure detection circuits within these circuits. The modules detect a triac failure and blow a fuse. By blowing the fuse the module has a higher probability of failing open, the most common fail-safe state.

Another failure detection technique uses an arrangement as shown in Figure A-9. Only one contact is held closed at a time. If the first triac fails, it is detected, its series contact is opened, and the output failure is reported. Now, the second triac can maintain control. If the second triac fails before the first triac is repaired, the fuse is blown.

One manufacturer markets an evolutionary I/O product that puts CPU processing power into the I/O modules. Each I/O module contains an on-board processor, extensive self-diagnostics, and communication capability over the manufacturer's proprietary control network. The system configuration consists of one or more PLC processors and one or more I/O modules linked together by a communications network. Each PLC processor on the network can be programmed to select the I/O it controls. In addition, the system allows for redundancy of PLC processors, communication media, and I/O modules. The self-diagnostics within the I/O are a key benefit to the system.

During each scan, discrete input modules may be checked for a failed switch, over-temperature, loss of I/O power, open wire, and input shorts. The diagnostic used depends on the type of input device and ranges from combination of the above four diagnostics to no diagnostics.

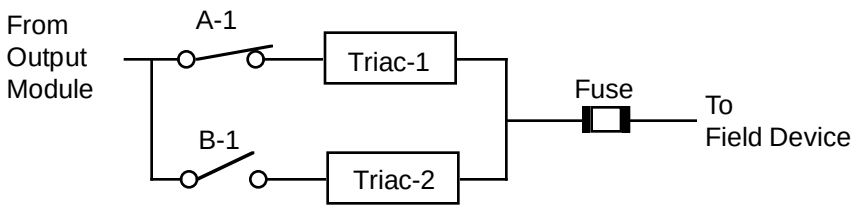


Figure A-9. A Failure Detection Technique Used in Output Modules.

During each scan, discrete output modules may be checked for a failed switch, over-temperature, loss of I/O power, no load, over load, short circuit, Load State Feedback, and a pulse test may be performed. Load State Feedback indicates the state of the output switch only, not the load. The pulse test routine uses a pulse signal to check proper operation of the output point. The pulse is fast enough to have no effect on most field output devices. The pulse test diagnostic is user-selectable. Depending on the type of output device the module receives different diagnostics. For example, 115 Vac isolated outputs receive all of the diagnostic types, while relay outputs have no diagnostics.

Analog input modules are diagnosed for under-range, over-range, open wire, high-alarm, low-alarm, intermittent fault, wire error, and input short. As in the other classes of modules, the diagnostics varies depending on the module type.

Analog output module diagnostics are under-range, over-range, and feedback error. Currently, under-range and over-range diagnostics are provided on all output modules, while feedback error is provided on only one module.

11.7. Infant Mortality

An important area of reliability that is often over looked is manufacturer testing. Infant mortality of hardware components in systems is a known failure problem. Infant mortality refers to the large amount of failures observed in a system during the early part of the system's life. Some key tests can be installed in the manufacturing process to reduce infant mortality. "Burn-in" testing is a big eliminator of components with short life spans. Some manufacturers burn-in at all levels of manufacturing: components, circuit cards, individual modules, and complete systems. Competent manufacturers maintain reliability groups that provide research on failure issues, testing of the PLC system, corrective action on all reported problems, and record keeping.

12. REDUNDANCY

A common feature in all redundant systems is a high level of self-diagnostics. Without diagnostics the redundant processors would not be able to detect when to switch over or shut down. The diagnostics should sense critical hardware failures, report the fault, and shutdown the I/O or PLC system. Also, non-critical hardware failures should be sensed, annunciated, and the user logic or operating system should intervene to provide proper control of the fault.

Many manufacturers offer different levels of redundancy and redundancy for various components of the system. Some of the common configurations of the various levels and components are discussed below.

12.1. PLC Processor

The first redundant processor configuration uses duplicate processors. A typical control scheme is to have one processor control the I/O modules, while the other is an active hot stand-by. A dedicated high-speed communication network, which will be called a processor-to-processor network, passes the latest I/O data between processors. In this configuration, the input image table is passed from the controlling processor to the stand-by processor. Then both processors execute identical user programs. At the end of each scan information is exchanged and comparisons are made. If it can be discerned that the controlling processor has failed the stand-by processor takes over control. If a failure is detected, but it is not possible to distinguish which processor failed, then the PLC system (i.e., both processors, all communications, and all I/O modules) is shut down and a processor failure is reported. The stand-by processor uses additional diagnostic programs to monitor the controlling processor and to initiate takeover upon a detected failure of the controlling processor. The I/O is controlled through a control network that allows multiple processors and multiple I/O on the network (multi-drop configuration is often used). Each of the three communication networks—the control network, system network, and the processor-to-processor network—are independent of each other. See Figure A-10 for a dual processor with single I/O.

Elevating redundancy to three processors brings in the advantage of two-out-of-three voting. The great advantage to two-out-of-three is that a fault in one processor does not stop control of the process. The triple redundant processor system is called Triple Processor with Single I/O and is shown in Figure A-11. As the name implies, this configuration has single I/O points and three processors. The input point is read by all processors. Usually, a buffer holds the data until all three processors acknowledge receipt of the input value. The processors execute identical or similar programs and send the output results to the voter. The voter takes a two-out-of-three vote and updates the output with the favored result.

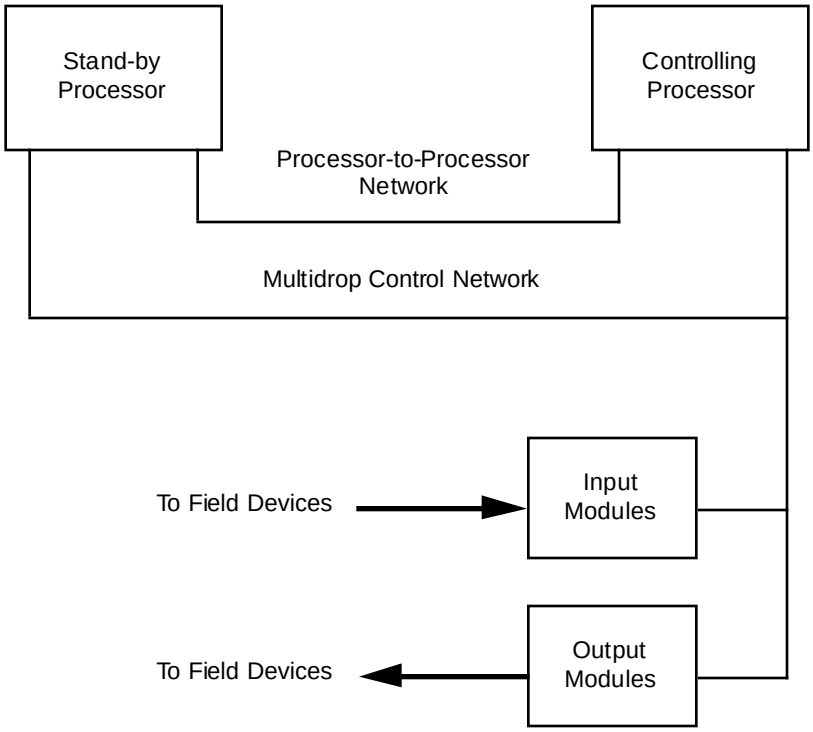


Figure A-10. Dual Processor with Single I/O.

In addition to the output being voted on, some triple processors vote on intermediate results within the user program. Via high speed communications, the processors share intermediate results. Checks and balances are designed into the runtime operating system to ensure the results go through two-out-of-three voting. This allows checking of intermediate results, such as in a cascade PID loop. A cascade PID loop uses two PID loops. The first loop receives the input (process variable) and outputs a set-point to the second loop. With intermediate voting, the first loop's output could be verified before passing it to the second loop.

12.2. I/O

Dual redundancy of I/O points is typically accomplished with an additional I/O module and the appropriate field wiring, rather than internal I/O module electronics. The various wiring configurations are discussed below. Triple redundancy I/O modules are common with manufactures of triple modular redundant, TMR, systems (discussed in Section 13).

Parallel wiring of discrete inputs yields PLC input module redundancy and increases the reliability of the PLC system. For parallel inputs requiring field power, the power supply must be selected, so it can supply enough current or voltage to drive the redundant inputs.

For most discrete output applications, a fail-safe condition is the output failing open. This alludes to a potential problem for output modules using triacs as the signal conversion device. Triacs have a high probability of failing short. To minimize the probability of triacs failing short, they can be wired in series. Now, both triacs must fail short to create a fail-to-danger condition. The fail-safe faults are: one triac fails open, one triac fails short, or both triacs fail open. The only fail-to-danger condition is both triacs failing short. This control strategy yields an increase in fail-safe faults and a considerable hardening to fail-to-danger faults. Parallel output wiring is desirable in limited cases where the output must fail short in order to fail-safe.

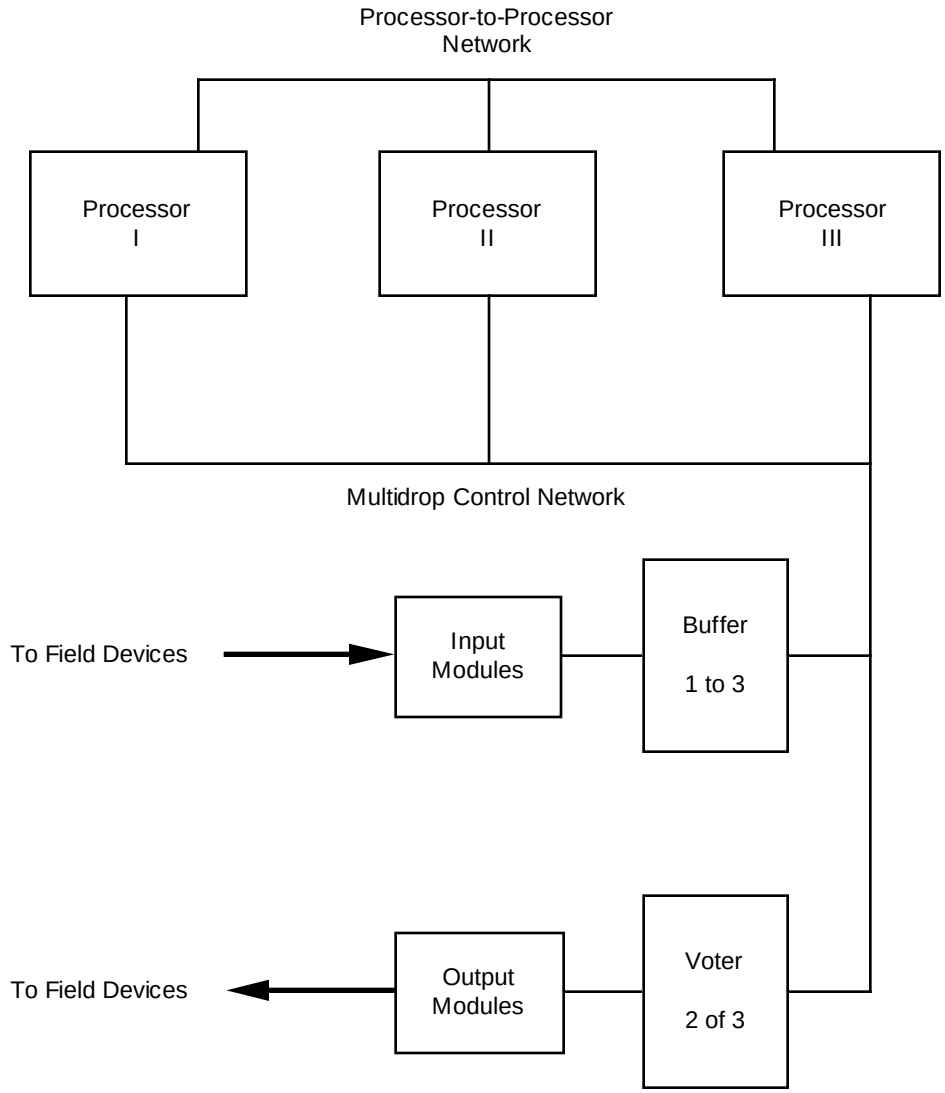


Figure A-11. Triple Processor with Single I/O.

For relay outputs, series wiring of normally open contacts is desirable for fail-safe conditions where the output should fail open. In the limited cases of failing short being a fail-safe condition, normally closed contacts would be wired in parallel.

The analog inputs may be wired in parallel provided the input signal can drive the two input modules. For analog output modules, module redundancy cannot be added. A redundant field device is required if analog output points are to be duplicated. Most analog output modules have diagnostics built in and can set the output to a desired value upon certain detected failures.

12.3. Communications

Any of the communication networks on the PLC system may have dual redundant capabilities. Various levels of redundancy are implemented in PLC communication systems, three of which are discussed below.

The first scheme uses redundant media and switches between media if a fault is detected. The processor verifies proper operation of the main system network. When a failure occurs the processor switches over to the backup network. This increases the availability of the network.

The second scheme is similar but does not use switching. Instead, the same message is sent out on both media. This scheme has the added benefit of using both messages for error handling routines, speeding up the data transmission rate by reducing the number of re-transmissions.

The third scheme uses the two redundant networks independently. The addressing, data transfer, and all communication on each network is totally independent, thereby doubling the transmission rate of the system when both networks are working. When one network fails, it is detected, and the system communicates with one network.

All three schemes can be elevated to higher levels of redundancy. Redundant levels greater than three may be provided in special cases, but none of the major PLC manufacturers advertise anything greater than triple redundancy.

13. FAULT-TOLERANT SYSTEMS

The concept of fault tolerance represents an effort to increase a system's availability despite a limited reliability. Fault-tolerant systems are highly available, although not necessarily reliable. Reliability refers to the probability of component failures, while availability refers to the probability that the system performs its desired function. Various components of a fault-tolerant system may fail while the system continues to perform the desired functions. Thus, the system is unreliable, because it needs constant maintenance, and highly available, because the fault-tolerant design allows the system to perform its desired functions even with a failure in the system. The following configurations are common in the PLC industry.

The first configuration, a dual processor with dual I/O, is shown in Figure A-12. Note that the control network may not be connected between the processors in this configuration. This connection depends on the manufacturer's implementation of the network. A multidrop configuration would allow connection of processors, while a loop configuration would not.

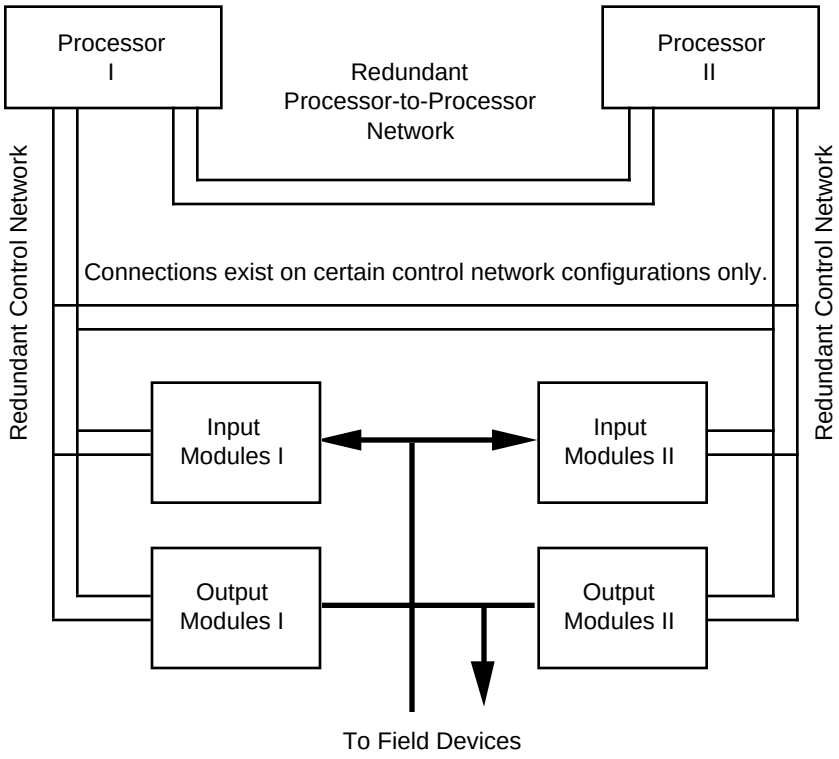


Figure A-12. Dual Processor with Dual I/O.

The two inputs are typically wired in parallel and either input can initiate a shutdown. The processors' output image tables are typically compared after each scan, and a mismatch causes the system to shut down. The outputs would be wired to increase the probability of failing safe. Most often, the outputs would be wired in series, but in very limited cases they would be wired in parallel. Any one of the processors, communication links, or I/O modules can fail, and the system should detect the failure, isolate the failure, and continue to operate. The primary disadvantage of this system is that a high number of fail-safe shutdowns occur. Anytime the two processors or the two inputs disagree and a failure has not been detected, the system is shut down.

Another configuration is triple processor with dual I/O. As previously mentioned, the advantage of three processors is that two-out-of-three voting can be implemented. There is one buffer for each input point and one voter for each output point. In addition, each buffer and voter contain diagnostics, fault detection, and some level of fault tolerance. The I/O modules are wired as in the dual processor with dual I/O system, but now when a processor fails or two processors disagree, the system continues to operate with redundancy. This system can continue to function with one processor, communication link, or I/O module failed. In addition, the nuisance fail-safe shutdowns caused by a dual processor configuration having processor disagreements are eliminated. Figure A-13 shows a triple processor with dual I/O configuration.

Taking the triple processor configuration one step further leads to the triple processor with triple I/O configuration. This is a common configuration, often called triple modular redundant (TMR). TMR allows two-out-of-three voting in the inputs, processor, and outputs. Three inputs are brought into each processor. Each processor executes the user program and outputs the results. Two-out-of-three voting occurs at each output point. This system, shown in Figure A-14, provides the highest availability.

Additional redundant levels may be utilized to improve the fault tolerance of the system. Currently, PLC manufacturers offer TMR systems as the highest level of fault tolerant systems. Two-out-of-four digital control systems are being designed in the control industry, and this may lead to PLC manufacturers using two-out-of-four configurations.

The processors are usually the most reliable piece of hardware in the PLC system. By providing redundancy in the processor, the availability of the system is only slightly increased. Most manufacturers are aware of this and offer redundancy down through the I/O modules. The following data was calculated using Markov models (Frederickson 1990). These data show

the dramatic increase in MTBF that redundancy and fault tolerance can provide. The MTBF for dual or triplicate processors with single I/O is four and five years, respectively. Add dual I/O to each processor configuration and the MTBF leaps to 26 and 61 years, respectively. With triple processors and triple I/O the MTBF skyrockets to 18,745 years.

14. PLC CLASSIFICATIONS

Since their inception, PLC systems have grown in size and complexity. Most manufacturers rate PLC systems by the maximum number of allowable discrete I/O points, which has lead to an informal size classification:

micro	<32	medium	256–1024
mini	32–128	large	1024–4096
small	128–256	super	>4096

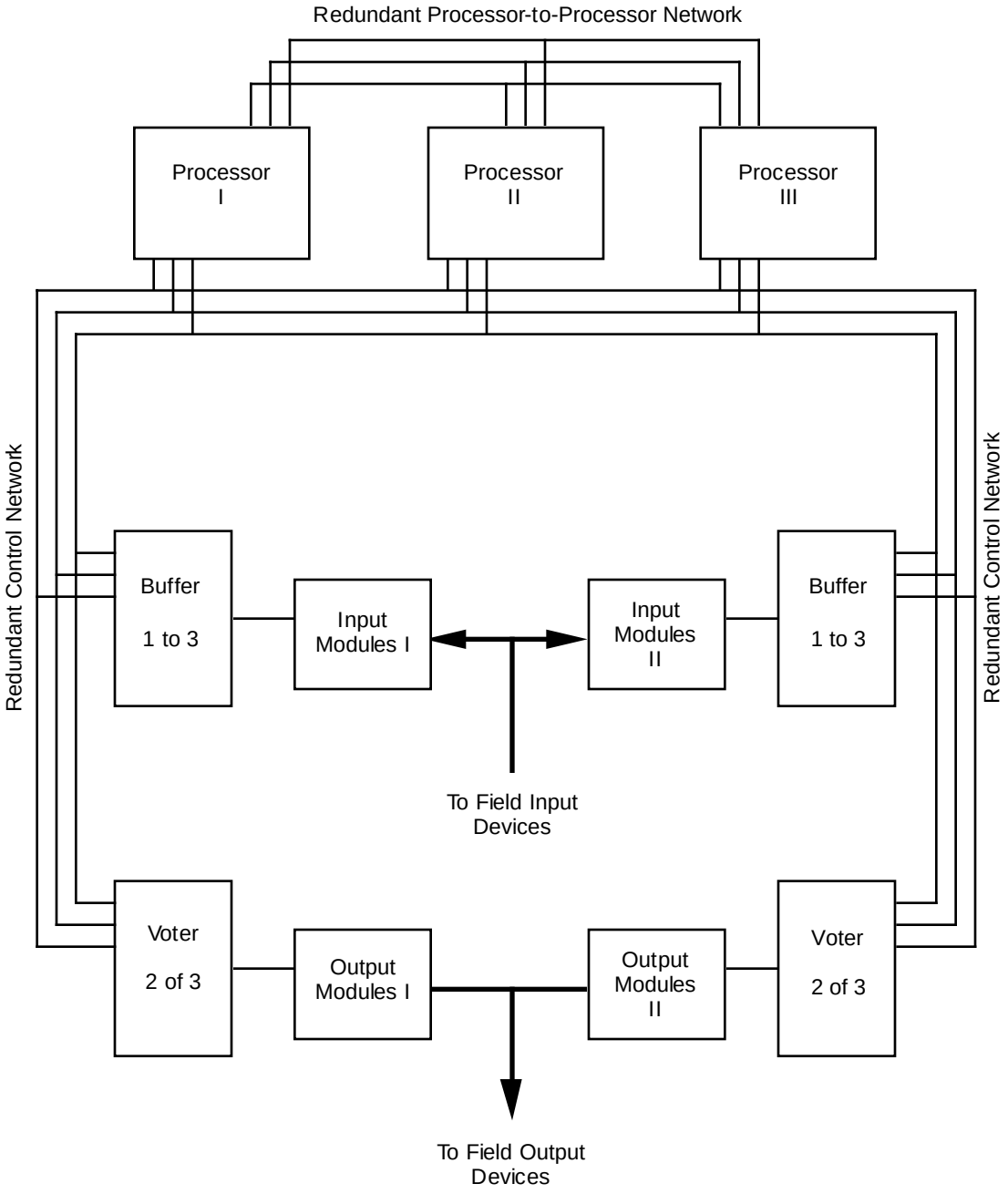


Figure A-13. Triple Processor with Dual I/O.

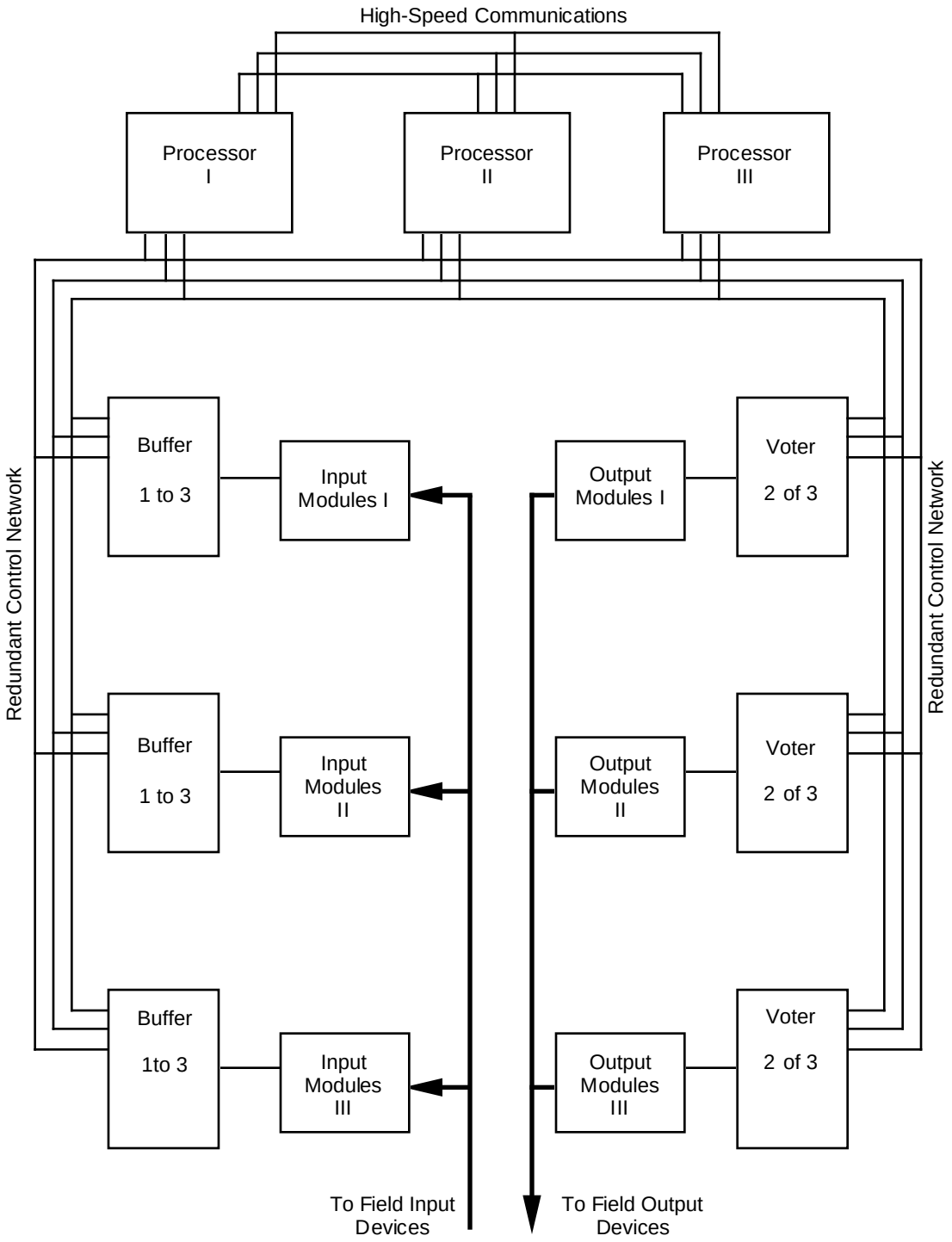


Figure A-14. Triple Processor with Triple I/O.

Appendix B:
Application of Programmable Logic Controllers in Safety Shutdown Systems

J. Palomar

R. Wyman

Manuscript Date: June 30, 1993

Contents

Figures

Appendix B:
Application of Programmable Logic Controllers in Safety Shutdown Systems

1. INTRODUCTION

This document provides an overview of the use of programmable logic controllers (PLCs) in emergency shutdown (ESD) systems. The intent is to familiarize the reader with the design, development, test, and maintenance phases of applying a PLC to an ESD system. Each phase is described in detail and information pertinent to the application of a PLC is pointed out.

2. SCOPE

ESD systems are required to be both fail-safe and highly available to mitigate the effects of accidents when they occur. PLC systems can be applied to meet both ESD requirements, but careful consideration to safety must be given throughout the life of the system. Because of their great flexibility, ease of interfacing, and cost-effectiveness, the PLC-based ESD system is gaining wide acceptance. There is, however, a great need for standards that directly address these applications. Currently, documents are being written by various experts and committees to address the usage of PESs in safety applications. This paper presents some of the ideas developed in those documents.

Most PLC applications are not publicized via trade journals or conferences. Of the few applications that are publicized, only a handful are ESD applications. More of the documentation is written to convey the design principles that should be employed when developing a PLC-based ESD system.

For this reason this document concentrates on PLC Life Cycle Issues. This discussion covers the design principles and examines techniques of project management, safety analysis, design, testing, installation, and maintenance over the life of the system. Section 5 offers several comparative design implementations, while section 6 highlights the important points discussed in first part of the document and provides some examples of PLCs in ESD systems. This document represents a consensus on which elements make up a safe and highly available system.

3. DOCUMENTATION

Throughout all phases of the life of a project one important element stands out: documentation. Documentation that reflects the current state of the system is an invaluable tool. Dennis A. Inverso (1991) points out:

“Documentation is defined as a vital, recorded information base used during all phases of developing and maintaining a Programmable Electronic System (PES).”

The development, use, and maintenance of a system’s documentation can dramatically improve the quality and safety of the system. Documentation obviously provides the foundation for understanding the system. In addition, the development and maintenance of the documentation facilitates communication among the various disciplines; operators, system designers, programmers, instrumentation and controls engineers, electrical power system engineers, and maintenance and management personnel. The documentation, then, provides a platform for intense scrutiny of the system. In Section 4, PLC Life Cycle Issues, the types of documents needed for each phase, the specific elements included in each document, and the importance of each document are detailed.

In any project, one of the initial steps should be defining the necessary documents. The initial list of documents does not need to be rigidly followed, but should be used as a tool to provide comprehensive information about the system. As the project progresses documents may be added or deleted and the definitions may change. But this list, like any other project document, should be maintained so that it reflects current thinking.

There are three main phases in producing and maintaining quality documentation; writing, reviewing, and updating. The documents must contain as much detail as necessary, while staying within the document's scope. Thus, the person writing the document must be proficient at the required discipline and he or she must also be capable of clear and concise writing. This point cannot be too highly emphasized—it is unimportant how much the author knows about the system if he or she cannot convey this understanding to others through the written word.

Care must be taken in selecting a review team for each document. Each review team should comprise competent persons from the appropriate disciplines who can analyze the functionality described in the document. Their concern should be with assuring the clarity and completeness of the documentation.

Finally, there must be a formal process for changing the documentation. Any time a project change is made all documents which might be affected must be reviewed to see if changes are needed. Once a document is changed a comprehensive review must be completed. Document maintenance over the life of a project is a critical activity.

4. PLC LIFE CYCLE ISSUES

The requirements for good configuration and project management of a PLC system are no different from the requirements of any ESD system. A brief discussion with references to standards is presented on configuration management. Next, safety analyses of PLC-based ESD systems are discussed. Finally, the remaining parts of this section address specific PLC issues in various phases of the project life cycle.

4.1. Project Management

All projects, including those that contain PLCs, can benefit from good project management. Several standards, including MIL-STD-1456A, MIL-STD-483A, IEEE-828, MIL-STD-1521B, IEEE-1042, MIL-STD-499A, and IEEE-1058.1, address this issue. Through implementation of these standards, the decisions and changes made over the life of the project can be well controlled and well documented.

MIL-STD-1456A addresses the top-level management of the project, which is the configuration management plan. This plan describes the methods and procedures to be used to manage the functional and physical characteristics of the project. It describes the documents necessary to define the roles of all personal associated with the project, to define the hardware configuration, to define the software configuration, to define all interfaces, and to provide traceability throughout the life of the project for reviews and audits.

MIL-STD-483A goes beyond MIL-STD-1456A and provides more detail on the contents of the elements of the configuration management plan. MIL-STD-483A details the implementation and documentation required for the following areas:

- Configuration Identification
- Configuration Management
- Configuration Control
- Configuration Audits
- Interface Controls
- Engineering Release Control
- Configuration Management Reports/Records.

Similar to MIL-STD-483A, IEEE-828 provides specific requirements for a software configuration management plan. Each section of the software configuration management plan is listed and described in detail. In addition, examples are presented to help clarify the contents of the plan.

MIL-STD-1521B describes, in detail, the requirements for reviews and audits throughout the life of the project. Figure B-1 below summarizes the contents of MIL-STD-1521B.

IEEE-1042 suggests ways to apply a configuration management plan. This standard interprets how to use IEEE-828, addresses the issues involved in establishing configuration management on a project, and presents sample plans that illustrate configuration management plans for various types of software based projects.

Establishing and documenting a project management scheme can aid in successful completion of a project. MIL-STD-499A addresses project management from a system engineering point of view. The standard defines how to establish an engineering effort and a System Engineering Management Plan. Specifically for software, IEEE-1058.1 defines the contents of a software project management plan.

4.2. Safety Analysis

The military standards mentioned above address the issues involved with project management. They do not address safety analysis. The safety analysis entails various safety studies completed and documented at various phases of the system's life. Some safety analysis tools are FMEA, fault tree analysis, and CMFA. A complete safety study should be completed after the System Design Review, the HWCI/SWCI Critical Design Reviews, the System Formal Qualification Review, and periodically throughout the life of the system. These reviews are identified in Figure B-1.

Maintaining and improving the safety level of a system is a continuous effort. A one-step safety analysis is insufficient, because any change may introduce a fail-to-danger fault. Therefore it is critical to document every problem discovered over the life of the system and the steps taken to correct the problem, including a safety analysis of the change. This documentation can provide a good foundation for renovations and changes, and for the design of future systems.

The safety analysis described below should occur after the System Design Review and should be verified after the HWCI/SWCI Critical Design Reviews, the System Formal Qualification Review, and routinely verified after final acceptance.

4.2.1. Safety Considerations

Any system which presents a hazard to life, the public, or the environment can be thought of as two distinct systems: (1) the system that does the main work of the system, and (2) the protection or ESD system that is called upon to make the overall system safe in the event of a malfunction of (1). If it is assumed that the failure rates of these two subsystems are independent, then the probability of the occurrence of a hazard that is unprotected is the product of the probability of a failure in (1), which creates the hazard, times the probability that the ESD system will be unavailable. Systems in which the control and ESD systems share components will require more complex dependent probability analysis. Thus, once the hazard rate of (1) is known, the necessary design using ultra-reliable components and redundancy can be done for the ESD system to bring the unprotected hazard rate down to the required level.

For any system that may present a hazard to life or the public safety, a philosophy on safety and environmental concerns should be written. Hazards should be identified and parameters for each quantified. Actions required of the ESD system should be specified for each hazard. This analysis is input into the system design. The safety parameters include probability of occurrence, time to respond before a catastrophe occurs, specific process control considerations to address the hazard, and accessibility of necessary equipment to mitigate the hazard. Along with documenting the hazards and parameters, all assumptions and explanations need to be documented. Enough justification should be written so a typical engineer can understand how and why the safety calculations and decisions were made.

A number of papers describe and reference various approaches to safety quantification, including Saudi Aramco 1990, Balls 1991a, Balls 1991b, Bryant 1976, Fisher 1990, Frederickson 1990, Gruhn 1991, Health and Safety Executive 1987, Inverso 1991, Magison 1979, Paques 1990, and Sparkman 1992. Without appropriate quantification there is no real foundation for the proper decisions.

Additional items which will improve the level of safety of a system are listed below:

- Shutdown devices should be easily accessible and clearly identified.
- The power-up sequence should force the system into a safe state before program initiation.
- The PLC should provide diagnostics to isolate internal and output module faults.
- The PLC should monitor auxiliary power supplies.
- There should be strict control of all ESD programs to prevent casual modification.
- Documentation should be kept up to date.
- Code should be modularized to simplify production and maintenance.
- Pulsed signals should be used to turn safety devices on and off.
- The system should be extensively tested, including testing with fault conditions.

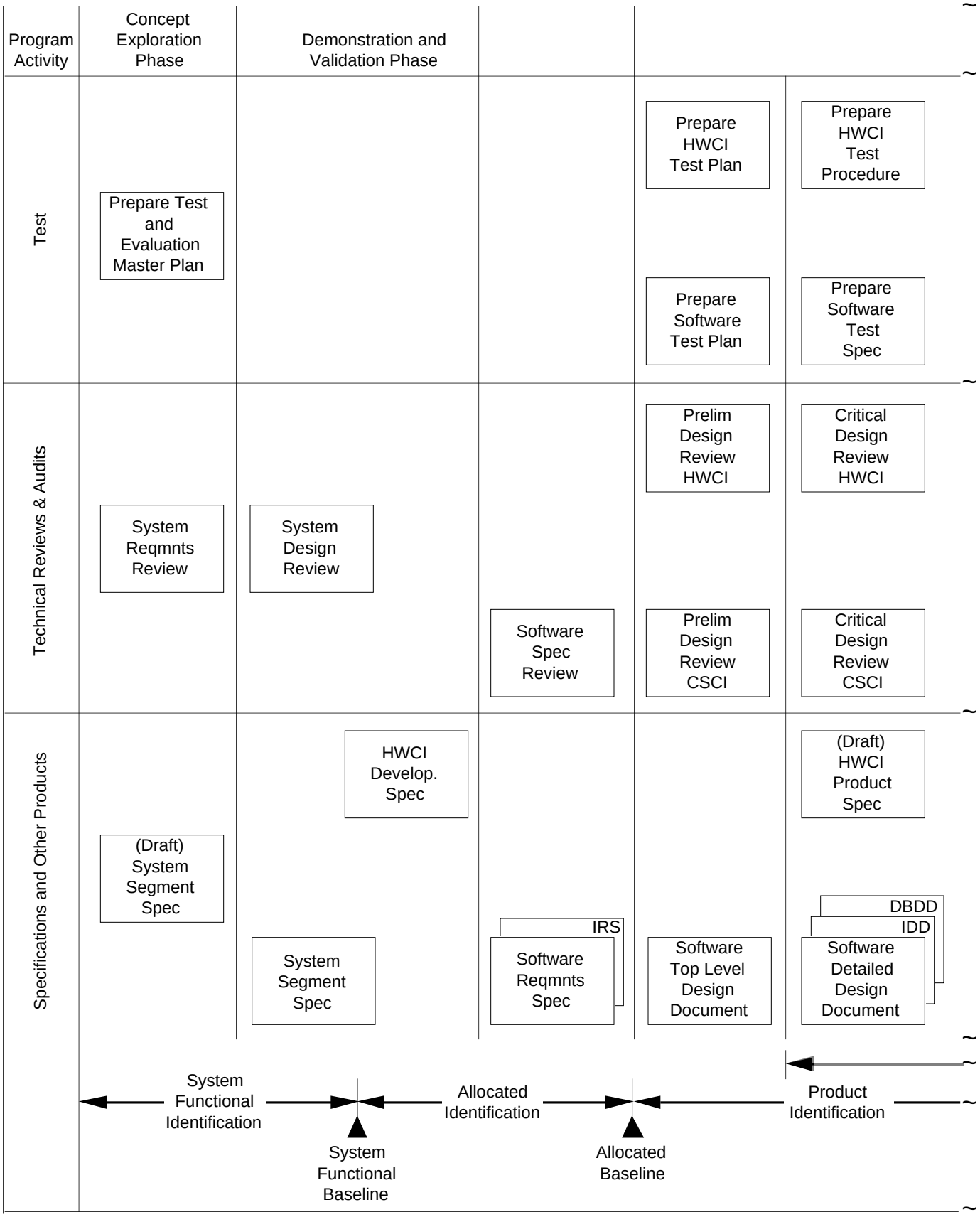


Figure B-1. Engineering and Test Flow Diagram.

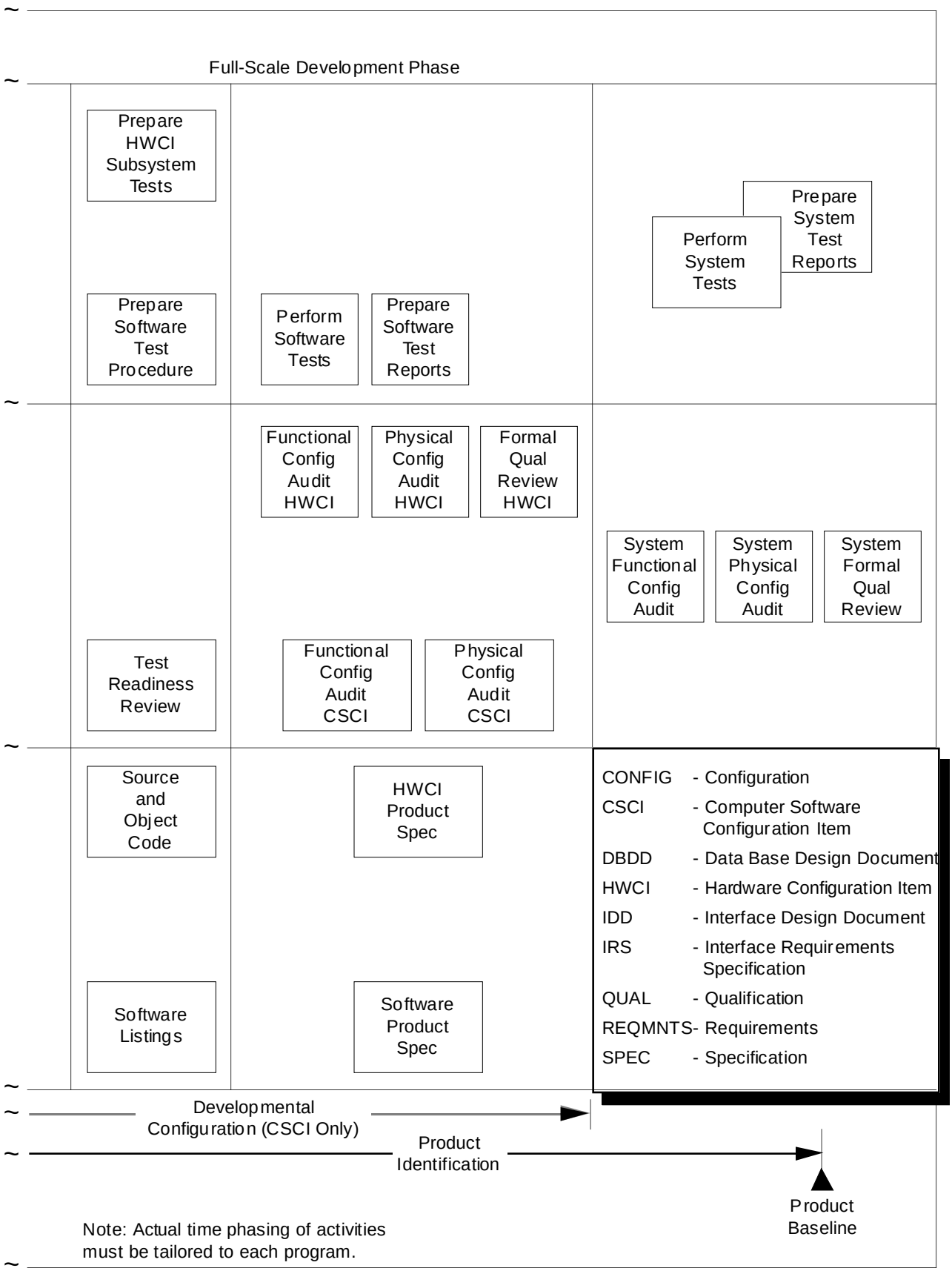


Figure B-1. Engineering and Test Flow Diagram (continued).

4.2.2. Availability

IEEE defines availability as the degree to which a system or component is operational and accessible when required for use. It is often expressed as a probability (IEEE). Availability is defined as:

A = Uptime / Total Time

where

A—Availability

Uptime—Time the system is working.

Total Time—Total mission time or time the system is needed to watch over the plant while the plant is in operation (excludes time for plant shutdowns).

or

A = MTBF / (MTBF + MDT)

where

MTBF—Mean time between failures.

MDT—Mean downtime.

These two definitions are essentially the same. See Balls 1991a, Balls 1991b, Fisher 1990, Gruhn 1991, Heron 1986, M. Smith 1991, and Walczak 1990 for more information.

MDT can be broken down into:

MDT = MTDF + MTTR

and

MTTR = MTDL + MTRF + MTRO

where

MTDF—Mean time to diagnose the presence of a system fault.

MTTR—Mean time to repair.

MTDL—Mean time to determine a fault location.

MTRF—Mean time to replace a faulted component.

MTRO—Mean time to return the system to operable condition.

The relationships between these various components can be seen in Figure B-2, below. Like reliability, the desired availability of the system should be clearly documented. Estimates of the various constituents of availability need to be made and the desired system availability needs to be determined. The acceptable availability will depend heavily on the desired hazard rate. Again, it is important to document all steps and decisions used to reach the availability number. Also, any changes should be documented, along with reasons why the changes were made.

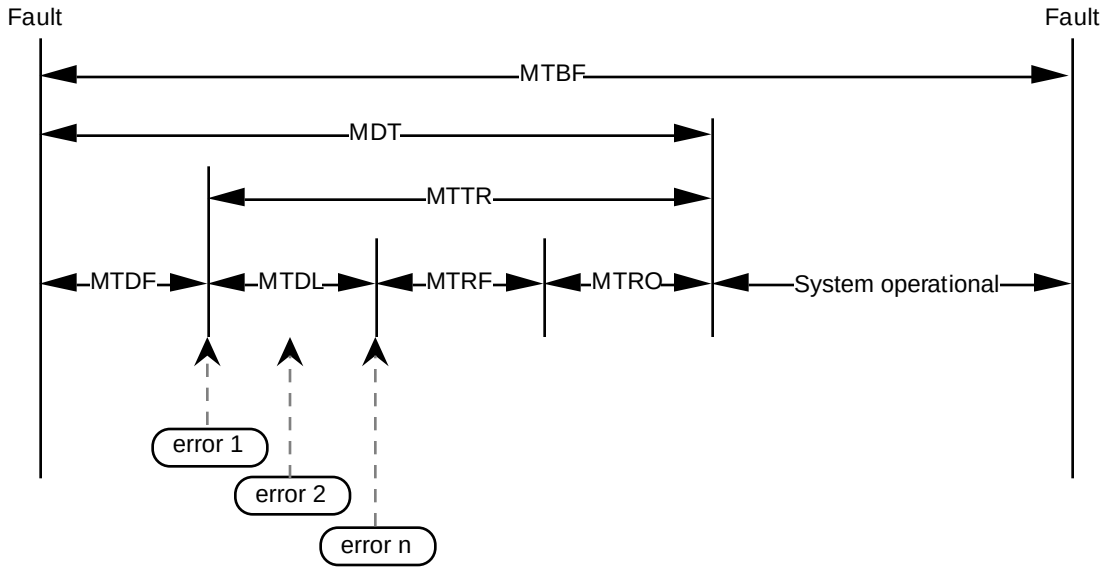


Figure B-2. Availability Components.

4.2.3. Hazard Rate

Hazard rate is the rate at which risks to life, the public, or the environment will occur. The hazard rate can be written as:

$H = D \times U$

where

H—hazard rate.

D—probable demand rate for system safety action.

U—unavailability (1 – A) of the ESD system, assuming that D and U are independent (Balls 1991a).

Hazard rate is the measure that relates the system availability and reliability to the concerns for safety. How low the hazard rate should be is determined by acceptable industry standards, the people at risk, and the plant managers. Of course, the hazard rate can not equal zero, but neither should the hazard rate be outrageously unacceptable by any of the groups mentioned above. The process of deciding what the acceptable hazard rate is and the means to achieve it should be clearly indicated in the documentation of the safety analysis.

4.3. The PLC-Based ESD System

Up to this point this paper has focused on issues pertinent to any ESD system. Below, this paper addresses concerns specific to PLC systems.

4.3.1. Hardware Selection

Hardware issues in a PLC-based ESD system are not very different from the issues needing consideration in any PES used in an ESD system. A few PLC-specific items are discussed below.

Once the configuration management and system requirements are detailed, the decision whether or not to use a PES may be made. Different systems should be evaluated, including PC, PLC, DCS, and SCADA systems. A thorough analysis of each system reviewed should be documented. Some scheme should be developed to show how each of the system requirements is met by each of the systems under consideration. It is particularly helpful if a numerical scoring system can be applied to this scheme so that the systems under consideration can be ranked according to their ability to meet the requirements, including reliability requirements.

The selection process must also include the consideration of the operating software that will be provided as part of the PES system under consideration. A vendor's method of collecting software fault reports and correcting software problems needs to be thoroughly understood and evaluated. If a vendor's software development process is visible, it should be evaluated to verify that the necessary QA/QC methods are in place to ensure that reliable software is being produced. And finally, field experience with the software being considered is an invaluable predictor of the kind of experience a new user may expect to have.

In making a selection, data from user groups can be very important. These groups are specialized to particular vendors and are formed to address system problems. They usually have considerable leverage with the vendor and can force necessary changes and guide future direction. A vigorous and active user group is usually an indicator of a successful product.

With the above data in mind, a choice can be made of the PES that best meets the needs of the system.

PLC systems provide a cost-effective solution to systems that have high number of I/O points. The major disadvantage of PLC systems is that they have poor user interfaces compared to PC, DCS, and SCADA systems. The PLC user interface typically consists of a programmer's terminal that displays a ladder logic image to be manipulated. Debugging tools allow the user to follow the flow of the power through the ladder logic image. Many of the special functions such as PID algorithms, math functions, and timers are difficult to implement, and process data can only be viewed with the programmer's terminal. Most PLC manufacturers are aware of the poor user interface attributes of their systems and are making efforts to improve them. Some items appearing on the market are graphical terminals to display process data, easier methods to program special functions, state-based programming, sequential function charts, and personal computers to provide better debugging, testing, and documentation.

4.3.2. Software Development

The reason for the development of PLC systems and other industrial PESs was to allow process control engineers and technicians to apply digital computer technology to process control applications, without having to learn the details of typical digital computer programming. Specialized languages and interpreters

were developed by the vendors, and these languages could be used by the process control engineer with very little training. PLC ladder logic is an example of such a language.

As these systems became more widely used, techniques were developed, usually by learning from bad experience, which tended to make the systems more reliable. Some of these techniques are described below.

4.3.3. PLC-Specific Considerations

The correct contact instruction must be selected for the desired circuit operation of the field device. The ladder logic program should be developed based on the PLC input points, not on the wiring configuration of the input devices. Two good rules to follow are:

(1) If the wired configuration of the field input device and the desired function in the circuit are the same, program the ladder logic input instruction as a normally open contact.

(2) If the wired configuration of the field input device and the desired function in the circuit are opposite, program the ladder logic input instruction as a normally closed contact.

The number of duplicate contact instructions should be reduced to a minimum, especially where a single contact instruction that is referenced to an existing coil instruction would suffice.

The use of positive feedback “seals” is recommended. An example of a positive feedback seal is a signal external to the PLC system which “seals in” a start switch to a motor. The project team should use the motor contactor auxiliary contact as a PLC input and the program should use this input to determine if the motor has started. The PLC output coil used to initiate a motor start should not also be used to verify the motor has started. In this way, the program indicates when the motor starts/stops, not when the motor has been requested to start/stop. Also, accidental restarts of the motor are reduced.

For critical applications, the use of auxiliary contacts such as those described above may not be adequate, since the auxiliary contact only indicates that the motor starter has picked up, not whether or not the motor has started. This is only one example of secondary sensing of a signal. Whenever a parameter must be sensed, the use of secondary sensing should be critically examined to determine what the consequences are if the secondary sensor indicates that the parameter is valid when, in fact, it is not valid.

The programmer should not program latched outputs that stay latched through power cycling. A holding circuit with “seal-in” contacts, commonly done with relays, should be implemented in the PLC program. This will eliminate the inadvertent turning on of motors, pumps, etc., when system power cycles.

4.4. Testing

Testing is a necessary part in the development of any ESD system. Testing may, however, turn up system deficiencies that require changes to be made in the implementation. Some of the changes are clearly benign, reflecting only errors in the implementation. These require changing the implementation, but the documentation is not changed since the documentation is correct.

Some changes, however, are a reflection of incorrect operation because of misunderstandings of requirements, or requirements or specifications that are in error. When such problems are detected, there should be a thorough review of the system since fundamental changes in the system design may introduce problems unless such reviews are held.

Finally, all changes should be documented, whatever the cause. Discrepancy reports should be issued together with resolutions so there is complete traceability of all of the changes made. Updating system documentation as changes are made is a critical activity since this is an area which can easily get out of hand in the rush to get a system into operation.

4.4.1. Hardware Test

The components of the system, the subsystems, and the complete system need to be tested. These tests may occur at various sites and be performed by various people. The key is not by whom or in what location the test is completed, but that the test is completed, documented and independently verified.

If the manufacturer is supplying a complete hardware system, then the manufacturer should be required to do all component and subsystem testing with the project team periodically reviewing the manufacturer’s tests and documentation. If the project team is purchasing and building the hardware, a team independent of the design and development team should complete all

testing, although this is not typical in the industry. In any case, all failures during testing should be well documented with the necessary safety parameters recorded (MTBF, MTTF, MTTR, etc.). These statistics can be invaluable in predicting ultimate reliability.

All hardware functionality should be tested. In addition, the hardware needs to be tested to ensure it meets the requirements of the hardware design and the system design. All interface specifications between the I/O modules and field devices should be verified. In addition, the I/O modules should be connected and exercised with loads having identical impedances to the field devices they will be operating.

4.4.2. Software Test

Typically, software is used to minimize hardware or to solve complex problems that are difficult to solve with hardware. The inherent complexity of software has led to a lively public debate on the question of how to verify that software is reliable, dependable, and safe. Although comprehensive discussion of the best way to test software is out of the scope of this paper, a few key ideas will be presented and some sources of additional information will be listed.

Although there is no way to “prove” that a software system is reliable, confidence can still be gained by various testing methods. Further details on software testing and metrics can be found in Humphrey 1989, IEC-880, Parnas 1991, Sandia 1987b, and Sparkman 1992.

All software used—including the software developed by the project team and that supplied by the PLC vendor—must be tested to some extent. The functionality of all pieces of software should be exercised and failure records should be kept. For the software that is produced outside the project team, reliability statistics should be obtained from the software manufacturer if possible.

One source of reliability statistics for purchased software is user groups. These groups usually have a vested interest in the success of particular software products and are only too happy to publicize problems, as this kind of publicity is a strong lever for forcing a software vendor to make necessary changes.

4.4.3. System Integration

If good practice has been followed to this point, the process of systems integration should have very few difficulties. The hardware and software should meet the system requirements defined earlier in the program, and any changes in hardware or software that affected the system requirements should have been resolved. The system integration difficulties should, for the most part, consist of overlooked items.

4.4.4. System Test

A PLC system can be purchased either as a complete system from one manufacture or as separate components to be put together on site by the project team. Either way the system needs to be thoroughly tested with all the PLC hardware, all the software (with no modifications for test purposes), actual field devices where practical, and simulators where the use of the actual field devices is not practical. The simulation hardware should duplicate the impedance, response times, and other pertinent characteristics of the field devices. If the PLC vendor is supplying the complete system, the system should first be checked at the vendor’s site. This way hardware changes are easy to implement and, typically, the project team will still have maximum control over the vendor’s performance (e.g., a major payment to the vendor is usually made after delivery).

The processor scan time desired should be verified against system requirements. This can be done by measurement or calculation.

Careful design that includes attention to system response time, coupled with thorough debugging, simulation, and verification, will reduce timing problems. To avoid common timing problems, sufficient time must be given for each mechanical device to settle to a steady state before starting the next step in a sequence. Careful design will also help to prevent the occurrence of “infinite loops,” which develop when one event unexpectedly triggers other events, and a circle of such events is created.

4.4.5. System Final Acceptance Test

The System Final Acceptance Test should require a complete run of all functional tests for a specified period of time with a maximum number of allowable hardware or software failures. This will provide a test on the MTBF requirements as well as functionality.

4.5. Installation

Proper installation is a critical factor in reliable operation of the hardware. Most PLC manufacturers

offer installation guidelines with detailed explanations outlining how to properly install their PLC systems. Items to consider are:

- I/O module to field device interface compatibility.
- Proper environment (temperature, humidity, air filtering, etc.).
- Industry standards and good practice for fabrication and wiring.
- Proper wire bundling and labeling.
- Proper connector labeling.
- Proper strain relief on all cables and wires.
- Proper grounding of all equipment.
- Avoidance of ground loops.
- Segregation of AC, DC, and communications wiring.
- Use of shielded cable for analog, pulse, or high-frequency signals.
- Use of secure connections (terminal blocks, twist lock connectors, etc.).
- Layout of the equipment for easy access to high-maintenance items.
- Proper lighting and space near equipment for maintenance personnel.
- Electrical transient protection where needed.

Following are additional items for which care should be taken during installation.

Output devices should not be connected in parallel for higher current carrying capacity. Output device specifications will not guarantee simultaneous switching for identical devices. Thus, if two identical devices are connected in parallel, the first one to turn on will be excessively stressed. Also, the “ON” resistance may be different in each device, causing the current division to be unequal. This may also cause excessive stress in one of the devices.

Connecting output devices in parallel to increase current capacity causes an abnormally high rate of failure (Wilhelm 1985). The proper design will use one switching device with the proper interface. If parallel output devices are desired for a fail-safe design, then each output device should be rated to properly handle the full load.

Inadequate protection against electrical transients can be a cause of random hardware failures. Following are some items that need electrical transient protection, as well as some protection suggestions.

Output modules and field devices with switches/mechanical contacts may need de-bouncing or surge suppression. Mechanical contacts bounce when closing. A properly designed circuit will filter out this bouncing and produce a clean transition.

An inductive DC load connected to an output module should have a diode in parallel with the load to handle the surge when the inductor is switched off. This surge will create a voltage spike that may damage the output module if there is no suppression mechanism. These diodes are commonly referred to as back-emf diodes and may be incorporated within the output module.

The proper interface design between the I/O module and the field devices is of utmost importance. Some output modules “leak” when in the OFF state, and if such a module is driving a high-impedance load the load may not turn off. Some input modules may have too high or low an impedance for the source and so operation may be erratic or improper. Also, input modules with high input impedances may be excessively noise-sensitive unless proper steps are taken.

The PLC system should have its own isolated power source and, if the system requires it, a UPS. Devices such as motors, generators, welding equipment, and heating devices, which tend to produce substantial noise on the power lines, should not be placed on the same electrical circuit as the PLC system. If properly designed, a voltage regulating transformer can be very effective. A separate receptacle on the same isolated electrical circuit should be provided for the PLC programming device. This receptacle should be properly designated for PLC use only and not be used for routine maintenance equipment (i.e., drills, electrical saws). Mounting the receptacle inside a locked PLC cabinet can prevent improper use of the receptacle.

The installation process should be a well thought out and well documented step-by-step procedure similar to the following example:

Phase 1—Install the system and verify correct installation.

Phase 2—Check supply circuits for proper wiring and voltage.

Phase 3—Energize each device, processor, I/O module, etc.

Phase 4—Energize all I/O circuits, from the modules to the field devices.

Phase 5—Perform a loop test of each control loop.

Phase 6—Operate the PLC inputs and outputs, with actual field devices where practical.

Phase 7—Installation complete, ready for final acceptance test.

4.6. Maintenance

The Maintenance discussion is split into two areas: hardware and software. Most PLC manufactures provide maintenance information. From this information the project team needs to develop maintenance procedures.

4.6.1. Hardware Maintenance

If the entire PLC ESD system is on a battery-powered UPS, these batteries need to be routinely checked and tested. The battery voltage needs to be checked for proper level, and a deep-cycle test should be performed routinely. Additionally, most PLC systems have battery back-up for volatile memory. The alarms for the each of these battery systems should annunciate locally, in the main control room, and in a central maintenance room if one exists.

A PLC system with software verification of the hardware I/O module arrangement will catch accidental insertion of an I/O module into the wrong slot. This piece of software, called “Traffic Cop,” is configured by the user. The configuration shows the mapping of I/O modules to PLC slots. Once the Traffic Cop configuration is set up the PLC processor routinely scans the modules and verifies that the I/O module configuration is identical to the Traffic Cop configuration. Not all PLC systems contain this feature.

Mechanical keying of I/O modules allows a specific type of module to be placed in each I/O slot. Typically the manufacturer designs each I/O module type with a unique hole pattern on the back, and the user sets the appropriate pin pattern on the slot. Thus, only one type of I/O module can make electrical connection in that slot. Each module type—digital input, analog input, digital output, etc.—will have a different hole/pin pattern. Not all PLC systems contain this feature.

“Hot-swap” is another convenient maintenance feature. Hot-swap allows replacement of any module (i.e., I/O, processor, communication, special-purpose) without damage to the module while the PLC system is under power. Not all PLC systems contain this feature.

When a failure occurs the system should be thoroughly diagnosed and all hardware problems found and repaired. It is in many cases easier to make adjustments to the software than to try to identify the source of a hardware problem. No changes to software should be allowed without management approval and appropriate documentation changes.

There should be in place and in force a system for documenting all failures of every type, including the time each failure occurred, what diagnostics were run and what the diagnostics found, and what was replaced as well as what actually fixed the problem. The time to repair the problem should be recorded, and if the failure was linked to any previous problems, this fact should also be recorded. From these numbers a true picture of system availability and reliability can be developed, together with identifications of weak points in the hardware.

Failures have two sources: (1) wear-out and breakage, and (2) design flaws. Type 1 failures require only replacement of equipment. Type 2 failures require considerable analysis and are covered in Section 4.7.

4.6.2. Software Maintenance

After start-up there are only a few software maintenance issues to consider such as securing back-up copies of the program and keeping track of the versions. Maintenance personnel may, with appropriate approvals, monitor the software during troubleshooting or use diagnostic software, but once the system software is installed, tested, and accepted, it should not be modified without formal approval. This approval must be preceded by a written analysis of the problem, a proposed solution, and a safety analysis to verify that the system will not be compromised by the change proposed.

Industry practice makes available to the user inexpensive hand held devices which allow set point modifications, forcing of I/O points, software changes, etc. These devices have a potential for misuse and careful control needs to be exercised by supervision to prevent mishaps.

From time to time, over the life of the system, the vendor of the operating software will send updates for the system or new versions of the system will be issued. Before any of these changes are made, the scope of the changes should be thoroughly understood as they pertain to the user's system. Finally, the changes should be installed and a thorough acceptance test of the system run. If any problems occur, the old system should be re-installed and run until the issues have been resolved.

A procedure must be established to control access to the software and assure that the correct version is running on the system with back-up copies available. One person, together with an alternate, needs to be assigned the responsibility for maintaining all programs and programming equipment. The alternate should be aware of system status (location of backup copies of the system and the code, current version running, etc.) so that he can handle problems in case the primary person is unavailable. Back-up copies should be stored under lock and key. One convenient technique is to place back-up copies in a box locked inside the PLC cabinet. Software access levels are good for controlling who has access to what functions. With the use of passwords, software access levels can be set up so certain people are allowed to program the PLC system, while others may only monitor the program and some may not be allowed to access the system at all.

4.7. Modifications

4.7.1. Hardware

Hardware modifications are the same for the PLC-based ESD system as for any other electronic system. If a hardware design problem is found, the appropriate procedures—such as those found in MIL-STD-483A and MIL-STD-1521B—are followed.

4.7.2. Software

Software faults occur during phases of design, development, and modification. The affects of software faults can be more subtle than hardware faults. Also, some software faults may not be detected for a considerable time.

In PLC systems, software modifications are much easier to implement than hardware changes, and for this reason there is a tendency for software changes to be implemented without reviews. In an ESD system it is important to follow a modification procedure, including thorough reviews, in order to verify that the change being proposed is the right change to make, as well as to reduce the number of new faults that may be introduced into the system by the modification. Changes should not compromise the safety function of the PLC.

Another feature which may have a detrimental effect on safety is "on-line programming." Most PLC system allow on-line programming changes by anyone who can access the PLC processor. The access can be through a hand-held terminal, a computer on the network, or a terminal connected directly to the PLC processor. Since the software is very complex and the threat to safety is real, on-line programming should be prohibited without proper managerial control. On-line programming devices must be strictly controlled.

5. COMPARATIVE DESIGN IMPLEMENTATIONS

To familiarize the reader with a PLC design, a simple Piping and Instrument Diagram (P&ID) is used to implement a logic diagram and wiring diagram. The PLC design follows the traditional steps of any instrumentation and control design. From a P&ID or some process/instrumentation diagram(s) the logic is formulated and a wiring diagram(s) is developed. A wiring diagram is required for all electrical designs, while the complexity of the P&ID motivates the decision to produce a logic diagram. The PLC wiring diagram is simpler than a system designed with non-computer electrical components and simply depicts I/O module connections to sensors and actuating devices. The PLC wiring diagram does not reveal the system logic. In a PLC, the logic resides in software and is commonly displayed to the user as ladder logic. (To further acquaint the reader with PLC systems, Figure B-8 shows various communication networks, I/O modules, and support devices that may exist in a PLC system.)

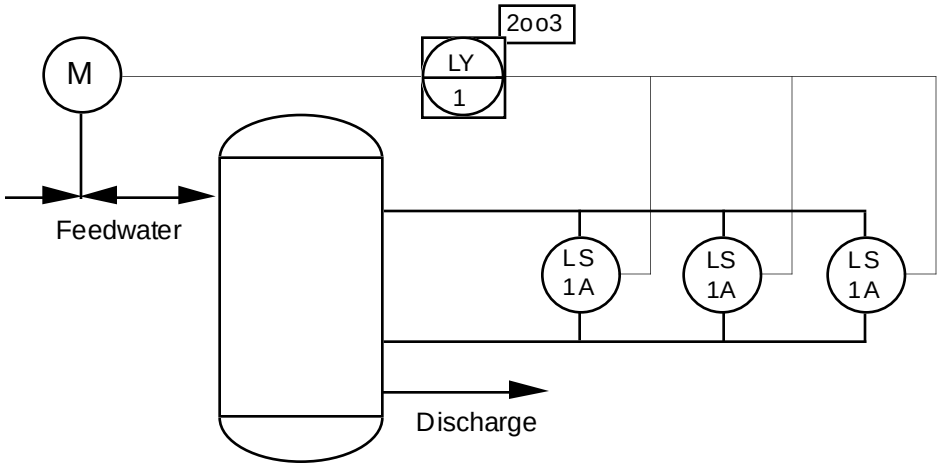


Figure B-3. Piping and Instrument Diagram.

Figure B-3 represents the Piping and Instrument Diagram (P&ID) that will be implemented with three different technologies. The three technologies examined are electrical, electronics, and programmable electronics. The electrical system is relay-based. The electronic system uses solid-state devices, and the programmable electronic system is a PLC design. For each technology the advantages and the disadvantages are listed and a wiring diagram is shown. The configuration logic is shown for the electronic and programmed electronic systems.

The P&ID depicts a system that will shut down feedwater to a tank upon detection of high water level. Three level sensors furnish signals for 2-out-of-3 voting and operator indication. If two out of three sensors detect a high level the voter initiates closure of the motor-operated valve, MOV. This example is based on a less-detailed presentation in Adamski 1991.

5.1. Electrical System—Relay Based

The wiring diagram for a relay-based implementation of this P&ID is shown in Figure B-4 below. Relay-based systems were the first protective systems installed and are still used extensively. The advantages and disadvantages of relay based systems are specified below:

Advantages:

- Easy to understand.
- High immunity to electrical noise.
- Highly predictable failure mode (typically > 90%).

Disadvantages:

- Require a lot of space.
- Complex functions are difficult or impossible to implement.
- Complex relay systems are difficult to troubleshoot.
- Large relay systems are difficult to maintain.
- All but small modifications are expensive, timing consuming and difficult.
- Typically designed to be energized under normal operation, thus require a significant amount of power to operate.
- Continued energization of the coil reduces the relays MTBF.
- Limited to digital I/O.

Referring to Figure B-4, a high water level will open the level switches LS-1A, 1B, and 1C. Opening any one of the level switches will turn on the High Level alarm light. Opening any two or more of the level switches will drop out the “D” coil and initiate closure of the MOV. Position switch ZS-1 shuts power off to the MOV when full close position is reached.

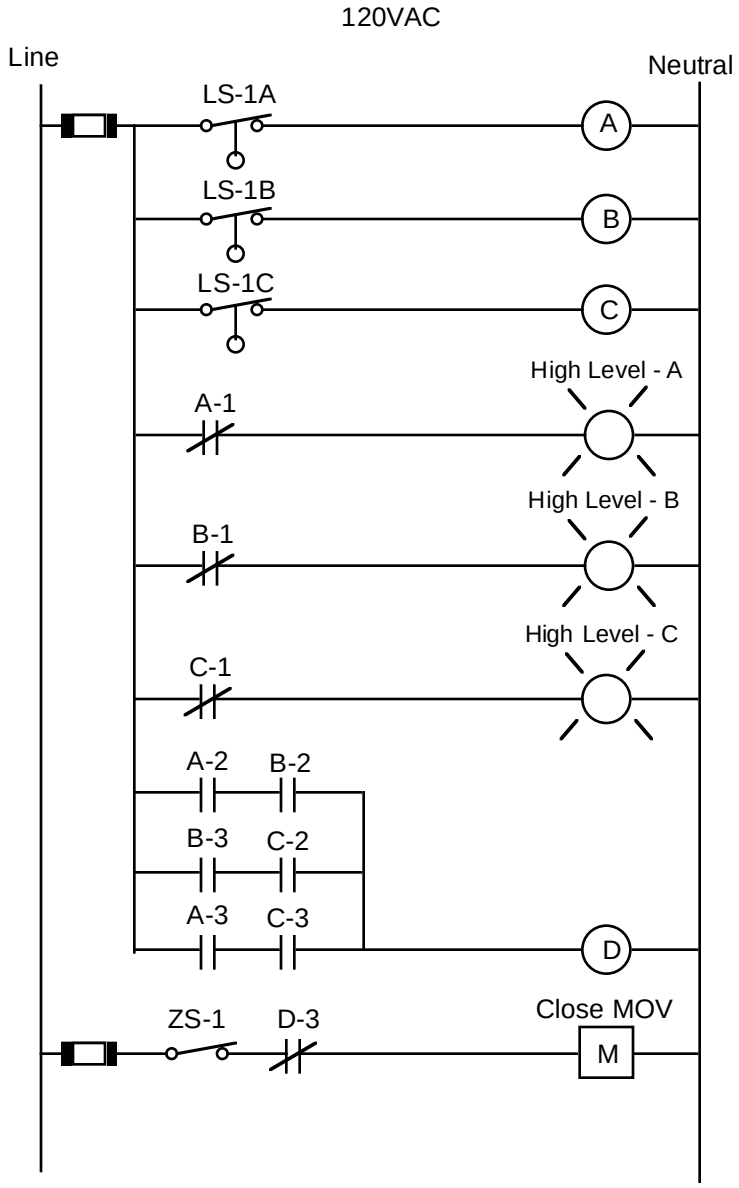


Figure B-4. Relay Wiring Diagram.

5.2. Electronic System—Solid-State Based

The wiring diagram for a solid-state-based implementation of the P&ID in Figure B-3 is shown in Figure B-5 below. The logic to be programmed in the solid-state device is shown in Figure B-6 below. Solid-state systems were used in limited cases and are rarely used now. The advantages and disadvantages of solid-state based systems are specified below:

Advantages:

- Small space requirements.
- Easy to test and troubleshoot.

Disadvantages:

- Failure mode closer to a 50–50 split.
- Limited diagnostics, typically LEDs connected to selected output points.
- Logic modifications are difficult.
- Obsolete hardware is difficult to obtain and maintain.
- Contains only digital I/O.

In the Solid-State Wiring Diagram (Figure B-5) the solid-state device inputs are on the right side and are labeled I1 through I4. A common neutral for all four inputs is shown. The solid-state device outputs are on the left side, are labeled O1 through O4, and have a common neutral as shown.

In the Solid-State Logic Diagram (Figure B-6), the inputs and outputs are designated as I1 through I4 and O1 through O4, respectively. Boolean logic operators are used to show the relationship between the inputs and outputs.

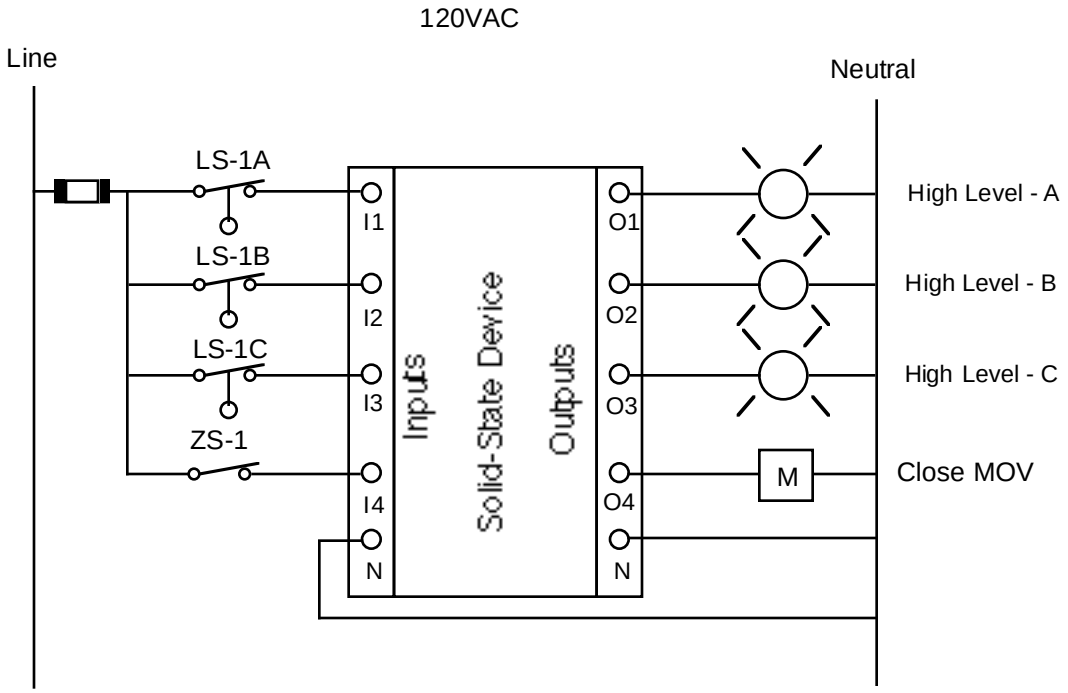


Figure B-5. Solid-State Wiring Diagram.

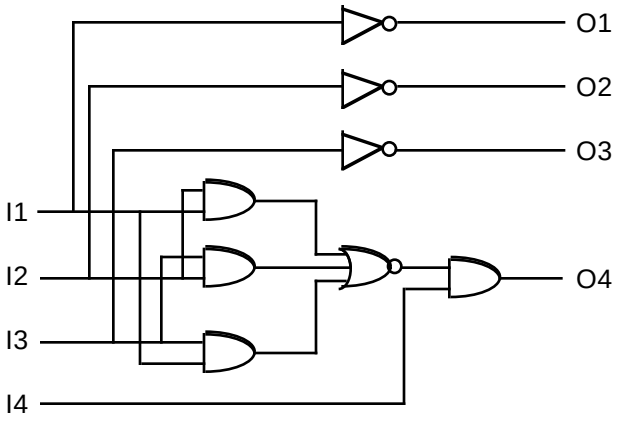


Figure B-6. Solid-State Logic Diagram.

5.3. Programmable Electronic System—PLC

A wiring diagram and ladder logic implementation of the P&ID in Figure B-3 is shown in Figures B-7 and B-8, respectively. In addition, Figure B-9 depicts the PLC and some of its components. The PLC has been widely accepted for control and protection system applications. Below are some advantages and disadvantages of a PLC-based system.

Advantages:

- Requires little space.
- Easy to test and troubleshoot.
- Contains analog I/O.
- Easy to modify both hardware and software.
- Complex functions can be implemented.
- Includes diagnostic routines.
- Contains communication links with other systems in the plant.

Disadvantages:

- Software reliability is difficult to prove.
- Difficult to control the hardware and software configuration.
- Expensive to manage the hardware and software configuration.
- The simplest software error may have catastrophic consequences (Vitrification plant (HM 1991)—a simple error, the omission of "+" in the program, failed to alarm a dangerous situation that could have resulted in the death of an operator.)

In the wiring diagram, the PLC inputs are represented by square boxes with the designation "I" and a number, while the outputs are represented by diamond boxes with the designation "O" and a number. Typically, the PLC I/O module internal wiring is arranged to divide the inputs or outputs into isolated groups of two or four that share a common neutral.

In the PLC Ladder Logic Diagram all contacts are shown on the right side and all coils are shown on the left side. The contacts controlled by I/O module inputs are designated by an "I" with a reference number. The coils that control I/O module outputs are designated by an "O" with a reference number. The coils and contacts internal to the PLC and implemented in software only are designated by "C" with a reference number. The internal coil and its associated contacts have the same reference number.

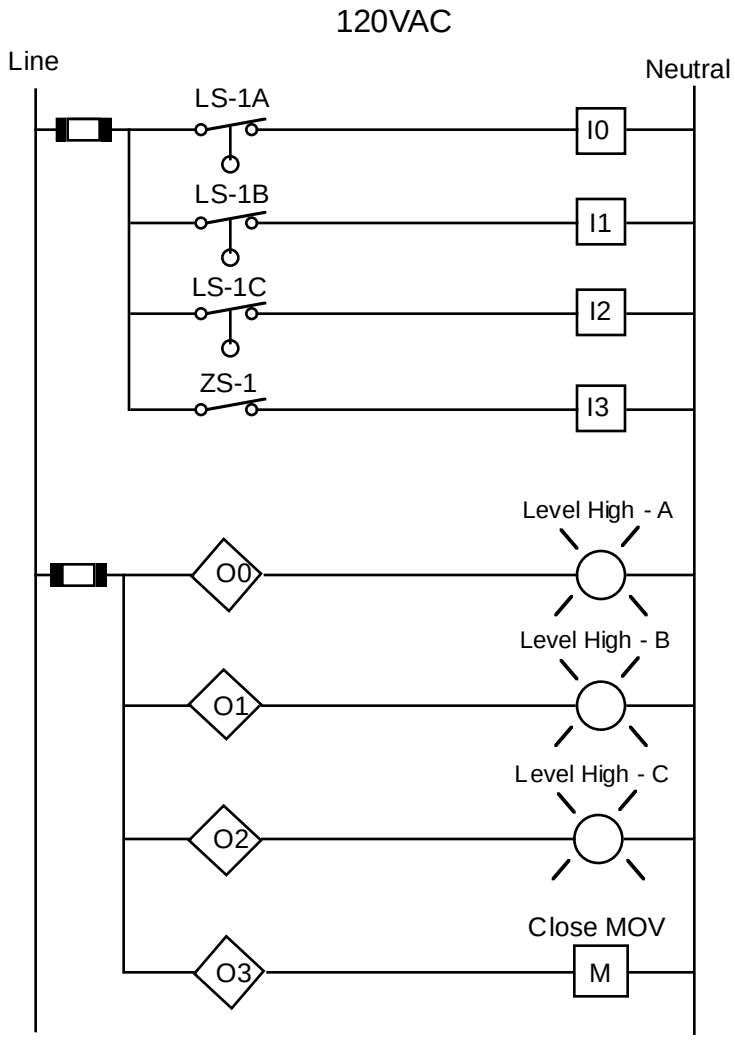


Figure B-7. PLC Wiring Diagram.

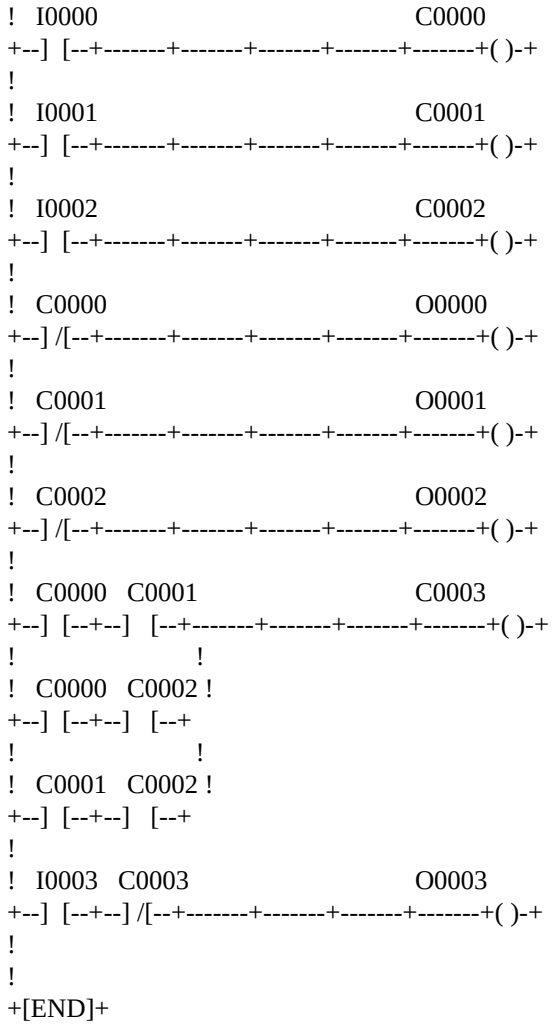


Figure B-8. PLC Ladder Logic Diagram.

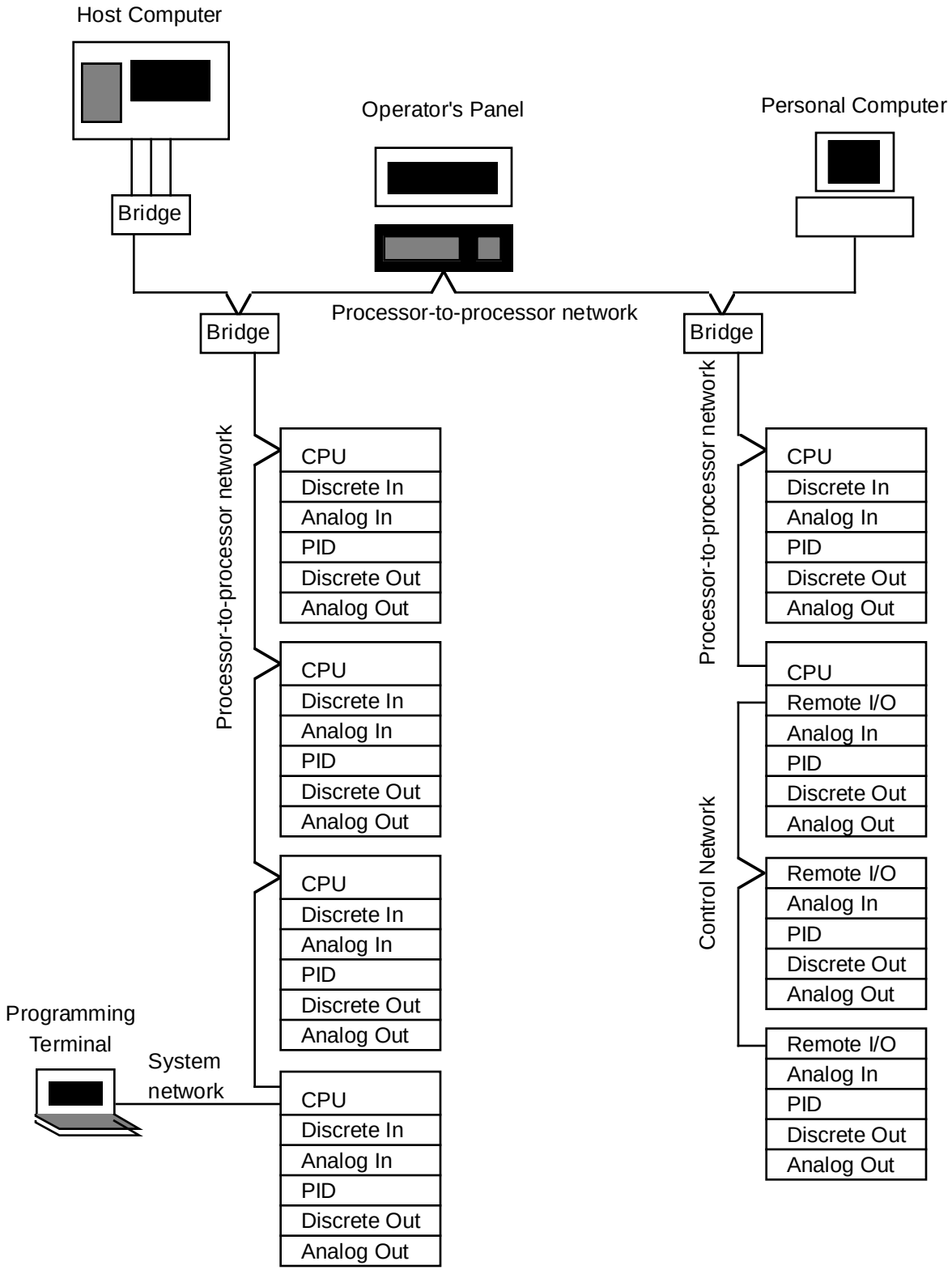


Figure B-9. PLC System.

6. APPLICATION EXAMPLES

The literature on applications of PLCs in ESD systems is very limited. Some documentation has been created on various applications of PLCs, but most of it does not go into much detail. Recently, more literature has been written on the use of PESs in ESD systems.

A few examples are presented and discussed in this section. Those examples address specific issues of PLC-based ESD systems. The pros and cons of the examples are discussed. Three papers that discuss key elements of the ESD system life cycle are also presented. The three papers address documentation, PLC qualification, and PLC safety.

6.1. All Side Construction, Inc.

Smith (1991a) discusses his application of a PLC system in an ESD system in conjunction with a DCS. The project was the renovation of a hard-wired relay-based shutdown system used in a petrochemical company's pilot plant. The paper details the areas of PLC selection, reliability, ESD design requirements, annunciator function, and shutdown bypass concepts.

Five PLC configurations were considered:

1. Single CPU system
2. Hot standby system
3. Duplex system
4. Triplex system with single sensors
5. Triplex system with triple sensors.

For various reasons, but principally cost, the single CPU system was selected. The single CPU system had an availability of 99.9947%, according to manufacturer's published literature, and various safety analyses completed at the design level concluded this was adequate.

A hazard and operability study (a qualitative analysis of the potential operational error scenarios and equipment failures) was performed. A more quantitative analysis in the form of Fault Tree Analyses was also performed at the design stage. The last consideration was checking for fail-safe operation of the ESD system.

The author puts forward some general principles for ESD system design which he feels are critical:

1. ESD system wiring should be separated from control system wiring so that a disaster that takes out one system's wiring will not take out both.
2. Giving the ESD system a distinctive color tends to prevent inadvertent modification of the system.
3. Critical valves (including manually operated valves) should have limit switches on them so that the ESD system can sense their position prior to system initiation in order to prevent system initiation if the valve is improperly positioned.
4. An uninterruptible power supply should be used where needed.
5. Lamp and horn test capability is required.
6. Regular test and maintenance are required.
7. Control and protection sensors should be separate from each other.

The author does not seem to be concerned about the software component of the system.

There have been no failures of any hardware components of the ESD system in the four years of operation. Smith states that the reliability of the system has been excellent, yet no safety statistics are quoted except hardware failure rate. The hardware reliability appears to have been excellent. No mention is made of the engineering change order record.

6.2. Flour Daniel, Inc.

Vora (1991) discusses some of the decisions that should be made by an engineering contractor once a successful bid for a control system has been won. The paper is biased toward the contractor's concerns of supplying a system that meets the requirements of the plant owner while allowing a profit to be made by the contractor. Still, a few good point are made about various key elements of PLC applications. Vora discusses the tradeoffs between PLC and non-PLC-based systems and goes on to discuss how to select the best PLC vendor. The PLC selection factors are listed in great detail and provide a good starting point for users selecting a PLC system. Another important area discussed is the future trends in the PLC market and the future needs of the PLC users.

Vora points out the great need for some standardization of the PLC communication networks, operator interfaces, and network addressing schemes. Also, diagnostics on most PLC systems need much improvement. The diagnostics can be made to reduce

the maintenance expertise and to help reduce the down time of the systems.

6.3. General Electric

Walczak (1990) lays out many of the features that he employed in using a PLC-based system for chemical plant protection. The chemical plant process involved highly flammable and toxic chemicals. One feature of the design process was the inclusion of the ESD system at the very beginning of the design. Because protection was built in from the beginning rather than added as an afterthought, the cost was reduced and the functionality increased.

Walczak (1990) states the project safety system objective very clearly (paraphrased):

> *The most important function of an emergency shutdown system is the ability to safely control and stop a process so that no injury will occur to personnel within the process area or any other associated area, provide protection for the plant and associated equipment, and to prevent pollution of the environment.*

During the design phase the system was divided into units of control, and each unit was assigned a safety level and priority. The response times required for safe operation were also determined.

The hardware requirements were developed with special consideration for system availability and the Single Failure Criterion from IEEE-279. To obtain the high availability desired, 99.99XX%, redundancy was necessary. Also, self-diagnostics were given high priority to help reduce the system down time.

A “hot spare” redundant system was employed with a synchronizing unit to keep the two units operating in step. Each PLC had its own I/O rack and each rack was connected to a redundant I/O bus. If the synchronizing unit discovered a failure in one PLC it signaled the other to take over operation, providing a “bumpless transfer.”

Considerable analysis was performed on the synchronizing unit because of its potential for being a single point of failure.

The response to various failures was analyzed, including the PLC system’s response to its own failures as well as external failures. The response of the balance of the system to PLC failures was also considered.

Considerable space in the paper was devoted to availability calculations, but no hard statistics were provided. The author devoted considerable effort to the question of what ought to be done to assure reliable hardware, without commenting on whether or not the vendor used actually performed the necessary environmental tests.

Too much time and effort was devoted in this paper to hardware considerations and almost no mention was made of software effort.

6.4. PLC Structured Programming

Keskar 1990 discusses a structured approach to programming ladder logic on a PLC. The system consisted of a PLC supervised by a PDP-11 computer. Because of the complexity of the supervisory control, a methodical and structured approach was decided upon to program the PLC.

Two major problems were identified. First, the communications between the PLC and the computer had to meet real-time performance requirements. Second, several engineers were involved in the effort to program the ladder logic.

The communications problem was not technologically challenging but only needed a complete set of documented requirements. Once the requirements were identified, the PLC system was designed to meet these requirements.

Two things were done to deal with the multiple programmer problem. First, the problem was minimized by the use of standard templates for implementing logic functions. The structured approach to programming ladder logic is highlighted in this article. This technique is equivalent to the use of a set of standard macros in programming. Second, standard names were given to all of the coils and registers to be used, and standard I/O assignments were made. In computer programming this is equivalent to constructing a set of global variables for all programmers to use.

Keskar outlines a simple approach to structured programming with ladder logic. This structured approach helped simplify the debugging of the complete software package, helped produce good

documentation of the software, gave some uniformity to the logic and program regardless of which person did the work, and made the logic easier to read. This example shows that a structured approach similar to that used in the computer science world can provide benefits to programming in ladder logic. A great need exists for formal methods and structured approaches like the one shown here, that deal specifically with programming languages used on PLCs.

6.5. PLC Qualification Testing

Ekkehard Pofahl ,of the Rheinland Technical Inspection Agency (TUV), has written a paper describing a series of tests which the agency performs to qualify PLCs for safety-relevant applications. However, the paper is badly translated and it is difficult read with any real understanding. Further investigations have uncovered a few more facts about TUV and its PLC qualification program. TUV is an independent organization that works closely with the German government. The organization certifies systems for use in safety applications (i.e., railroad, aircraft, nuclear, petrochemical industries). TUV has different classes of certification, and the German government requires a specific class of TUV certification for these safety applications. The organization is divided into various companies that appear to be named after the geographical location they service. TUV–Rheinland is developing a series of certification tests for PLCs. In addition, certain PLC manufacturers, actively selling their systems in safety application, are pursuing TUV–Rhienland certification of those PLC systems they expect to be used in German safety applications.

6.6. PLC Software Bug

Following is the text of a memorandum that was distributed when a software error was found in a PLC software product. The software manufacturer is referred to as SM, the PLC manufacturer is referred to as PM, and all names and phone numbers are suppressed in the copied version below:

A Bug in SM PLC Software (Version X.X) Can Cause Erratic and Unsafe Equipment Operation

THIS IS AN IMPORTANT SAFETY ISSUE—PLEASE ACT PROMPTLY!

Required Action:

If you are using a PM programmable controller and have programmed it with the SM programmable logic controller (PLC) using Version X.X software, please immediately reload your system using SM's latest software release: Version X.X. SM will supply you with the upgraded software at no charge.

The local sales representative for the software is name at distributor, phone number. For further information about this, please contact name, phone number, or name at SM, phone number.

The Problem:

SM contacted company to inform us that there is a problem in the Version X.X programming software for their Programmable Logic Controller. A bug in their product can corrupt a controller's logic and cause equipment to operate erratically. IF your controller operates such safety-related equipment as a laser shutter or an access door, this unpredictably erratic behavior can cause potentially hazardous safety problems.

Your cooperation is appreciated.

Name

Title, phone number

The problem that required urgent attention was a simple programming error. The programmer of the PLC software tool had failed to reset a pointer, and the pointer was vital to operation of the PLC upon downloading the new application software.

6.7. PLC Safety Concerns

To give the reader an idea of what level of redundancy and availability is needed to reach the safety levels required by industry, the example below is presented. It is assumed that the plant requires a hazard rate of 1 failure every 3000 years—a common requirement in the chemical, oil, or nuclear power industries.

The MTBF for the system failing dangerously was used to calculate unavailability. Markov models were used to calculate the MTBF for system failures (Frederickson 1990; Frederickson and Beckman 1991; Triconex 1990). Both the MTBFs of safe failures and dangerous failures were calculated. Safe failures are failures of the ESD system that cause the ESD system to place one or more field devices in a safe state. The safe state was taken as the de-energized state in the Markov models. Dangerous failures are failures of the ESD system that cause the ESD system to place one or more field devices in a dangerous state. The dangerous state was the energized state in the Markov models. For this example, the MTBF for dangerous failures will be used. The MTBF for safe failures is not considered since these failures do not create a hazard.

The MTBFs for dangerous failures for various PLC systems are listed in Table 1 below (Fredrickson 1990).

Table 1. Dangerous Failure MTBF of Various PLC Systems.

PLC SYSTEM	Dual PLC Single I/O	Dual PLC Dual I/O	Triple PLC Single I/O	Triple PLC Dual I/O	TMR PLC
MTBF DANGEROUS (years)	4.31	25.54	5.07	60.8	18,745

Table 1 assumes the following numbers:

MTRO = 0.5 hrs. An average of 0.5 hours is required to bring a piece of equipment back on-line.

MTRF = 1.0 hrs. The component is assumed to be received from stock within one hour.

MTDL = 2.5 hrs. The maintenance personnel require an average 2.5 hours to locate the fault. This includes the time from the initial detection of the fault, finding the correct solution, and waiting for a new part.

Totalling these figures yields a mean time to repair of

MTTR = 4.0 hrs.

The mean time to diagnose the presence of a system fault, MTDF, is related to the average time between system tests. If system test are completed weekly then the MTDF is 84 hours (assuming random failures may occur at anytime during the week) giving an MDT of 88.0 hrs.

The availability, A, (considering dangerous failures only) for each PLC system is calculated by

A = MTBF / (MTBF + MDT).

The results are shown in Table 2.

Table 2. Availability of Various PLC Systems.

PLC SYSTEM	Dual PLC Single I/O	Dual PLC Dual I/O	Triple PLC Single I/O	Triple PLC Dual I/O	TMR PLC
AVAILABILITY (%)	99.76746	99.96068	99.80225	99.98348	99.99995

Table 2 assumes the demand rate, D, of the ESD system is 3 demands per year, or

$D = 3.425 \times 10^{-4}$ demands/hour.

Now, the hazard rate, H, is calculated from

$H = D \times U$

where

$U = 1 – A$.

Table 3 shows the hazard rates for each system.

Table 3. Hazard Rate of Various PLC Systems.

PLC SYSTEM	Dual PLC Single I/O	Dual PLC Dual I/O	Triple PLC Single I/O	Triple PLC Dual I/O	TMR PLC
HAZARD RATE (hazards/3000 yrs.)	20.93	3.54	17.80	1.49	0.0048

As the above exercise illustrates, only one system meets the requirement desired, and that is a triple module redundant (TMR) PLC system. The TMR system is nearly three orders of magnitude safer than all other PLC systems. It is important to have redundant CPU modules, but the effect of redundant I/O modules is significant. From single to dual I/O modules nearly an order of magnitude of safety is gained.

The analysis here was completed for the PLC system only. The ESD project should perform a similar analysis for the entire ESD system, including the PLC, software, and field devices.

The amount of automatic testing on the system has a significant effect on down time, which can affect the hazard rate. Thus, the automatic testing of the system can be increased from weekly to daily and the hazard rates in Table 3 would improve. The amount of automatic testing to use is a trade-off between personnel effort and the desired safety level. The period between automatic testing can be made an insignificant factor in the safety equation if good preliminary calculations are made and the appropriate system is chosen.

Using the example calculations above, it appears a single TMR PLC system can meet the needs of the nuclear power industry. It is not clear, however, that the TMR system meets all other regulations and guidance of the NRC.

One issue to consider is the single-failure criterion (IEEE-279). Most TMR systems use identical hardware and software modules, thus opening themselves up to common-mode failures. Certain common-mode failures can fall within the scope of the single-failure criterion, thus requiring two TMR systems.

With quadruply redundant and completely independent safety systems, a single PLC with single I/O for each quadrant should suffice. Of course, the appropriate calculations should be provided to verify this assumption.

7. REMARKS

The life cycle of the ESD system must have a well thought out plan, and the plan must be flexible enough to allow changes without relaxing the safety requirements. The safety of the ESD system must be analyzed throughout the life of the system. It is not enough to analyze the safety of the system one time after the design is complete, because many changes to the system will occur after the completion of the design. Thus, it is important to verify the safety of the system after each change. It is also helpful to initiate safety audits that re-analyze the entire system at periodic intervals.

Documentation is very important. Good documentation of decisions and the reasons for the decisions, recorded throughout the life of the system, can reduce the number of errors introduced into the system. The project mangers must provide the necessary people or allow the necessary time to produce good documents. The documentation of the project will be invaluable when troubleshooting, understanding problems, training personnel, and aiding in avoidance and analysis of plant accidents.

Programmable PLC systems used in safety applications have many of the same issues and vulnerabilities as other software-based systems. Careful consideration of the use of standard methods of software review for PLC safety systems should be made. If such methods are applicable, then these methods should be embraced by the PLC community. It may be possible to identify a subset of those methods that would apply to PLCs specifically, but it would have to be shown that the subset was complete in the sense that the methods which were outside the subset did not apply to PLC software.

Appendix B

References

Adamski, Robert S., 1991, "Evolution of Protective Systems in the Petro-Chemical Industry," Instrument Society of America (ISA) Transactions—Control Systems Safety, Vol. 30, No. 1.

Balls, Basil W., and Cole, Simon R., "Design Principles for Safety Systems," Instrument Society of America (ISA) Transactions—Control Systems Safety, Vol. 30, No. 1, 1991a.

Balls, Basil W., and Gruhn Paul, "Design Considerations for High-Risk Safety Systems," Instrument Society of America (ISA) Transactions—Control Systems Safety, Vol. 30, No. 1, 1991b.

Banks, William W. Jr., and Wood, Charles Cresson, "Human Error: A Major Consideration in Systems Security," Lawrence Livermore National Laboratory, Document Number UCRL-JC-108922, Livermore, California, November 25, 1991.

Barter, Robert, and Zucconi, Lin, "Verification and Validation Techniques and Auditing Criteria for Critical System-Control Software," Livermore Lawrence National Laboratory, Livermore, California, September 1993.

Bongarra, James P. Jr, VanCott, Harold P., Pain, Richard F., Peterson, L. Rolf, and Wallace, Ronald I., "Human Factors Design Guidelines for Maintainability of Department of Energy Nuclear Facilities," Lawrence Livermore National Laboratory, Document Number UCRL-15673, Livermore, California, June 18, 1985.

Bowman, William C., Archinoff, Glenn H., Raina, Vijay M., Tremaine, David R., and Leveson, Nancy G., "An Application of Fault Tree Analysis to Safety Critical Software at Ontario Hydro," Ontario Hydro, Toronto, Ontario, Canada, M5G 1X6, no date.

Bryant, J.A., "Design of Fail-Safe Control Systems," *Power*, January, 1976.

DOD, See U.S. Department of Defense.

EPRI, *Procedures for Treating Common Cause Failures in Safety and Reliability Studies*, Volume 1—*Procedural Framework and Examples,* and Volume 2—*Analytic Background and Techniques*, EPRI NP-5613, 1988.

Fisher, Thomas G., "Are Programmable Controllers Suitable for Emergency Shutdown Systems?" Instrument Society of America (ISA) Transactions—Programmable Controllers, Vol. 29, No. 2, 1990.

Frederickson, A. Anton, "Fault Tolerant Programmable Controllers for Safety Systems," Instrument Society of America (ISA) Transactions, Vol. 29, No. 2, 1990.

Frederickson, Tony, and Beckman, Lawrence V., "Comparison of Fault Tolerant Controllers Used in Safety Applications," Instrument Society of America (ISA) Transactions—Control Systems Safety, Vol. 30, No. 1, 1991.

Gruhn Paul, "The Pros and Cons of Qualitative & Quantitative Analysis of Safety Systems," Instrument Society of America (ISA) Transactions—Control Systems Safety, Vol. 30, No. 1, 1991.

Hamacher, Carl V., Vranesic, Zvonko G., and Zaky, Safwat G., *Computer Organization*, 3rd ed., McGraw-Hill, New York, NY, 1990.

Hamming, R. W., *Coding and Information Theory*, 2nd ed., Prentice-Hall, Englewood Cliffs, NJ, 1986.

Health and Safety Executive, "Programmable Electronic Systems in Safety Related Applications: Part 1—An Introductory Guide," Library and Information Services, Broad Lane, Sheffield S3 7HQ, United Kingdom, 1987.

Heron, R.L., "Critical Fault Control System," Presented at Florida Municipal Utilities Association 21st Annual Engineering & Operations Workshop, Ft. Pierce, Florida, November 4-6, 1986.

HM Nuclear Installations Inspectorate, "Windscale Vitrification Plant Shield Door Incident 15 September 1991," London: HMSO, 1991.

Hughes, Thomas A., *Programmable Controllers*, Research Triangle Park, North Carolina: Instrument Society of America, 1989.

Humphrey, Watts S., *Managing the Software Process*, Menlo Park, California: Addison–Wesley Publishing Company, Inc., 1989.

Ichiyen, N., Clark, A.B., Joannou, P.K., Harauz, J., and Tremaine, D.R., “The Canadian Nuclear Industry’s Initiative in Real-Time Software Engineering.”

IEC, See International Electrotechnical Commission.

IEEE, See Institute of Electrical and Electronics Engineers.

Institute of Electrical and Electronics Engineers Inc., 345 East 47th Street, New York, NY.
IEEE-1042, “IEEE Guide to Software Configuration Management,” September 1987.
IEEE-1058.1, “IEEE Standard for Software Project Management Plans,” December 1987.
IEEE-279, “Criteria for Protection Systems for Nuclear Power Generating Stations,” 1971.
IEEE-352, “IEEE Guide for General Principles of Reliability Analysis of Nuclear Power Generating Station Safety Systems,” 1987.
IEEE-610.12, “IEEE Standard Glossary of Software Engineering Terminology,” February 1991.
IEEE-828, “IEEE Standard for Software Configuration Management Plans,” June 1983.
IEEE/ANS P-7-4.3.2, “Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations—Draft 4,” June 1992.

Instrument Society of America
ISA RP55.1, “Hardware Testing of Digital Process Computers,” 1983.
ISA SP84.02, “SP84 Programmable Electronic Systems for Use in Safety Applications, Annex K—System Analysis,” Draft 9A, July 1992.

International Electrotechnical Commission, 3 Rue de Varembe, Geneve, Switzerland.
IEC-65A, “Software for Computers in the Application of Industrial Safety-Related Systems,” August 1, 1991.
IEC-848, “Graphcet,” 1986.
IEC-880, “Software for Computers in the Safety Systems of Nuclear Power Stations,” 1986.

Inverso, Dennis A., “Document for Programmable Electronics Systems: Safety Considerations in Documentation for BPCS and ESDs,” Instrument Society of America (ISA), Paper #91-0333, 1991.

ISA, See Instrument Society of America.

Keskar, P.Y., “Structured Approach in PLC Programming for Water/Waste Water Applications,” Instrument Society of America (ISA) Transactions, Vol. 29, No. 2, 1990.

Lawrence, Dennis, “Software Reliability and Safety in Nuclear Reactor Protection Systems,” Livermore Lawrence National Laboratory, Livermore, California, June 1993.

Leveson, Nancy G., and Harvey, Peter R., “Analyzing Software Safety,” IEEE Transactions on Software Engineering, Vol. Se-9, No. 5, September 1983.

Leveson, Nancy G., Cha, Stephen S., and Shimeall, Timothy J., “Safety Verification of ADA Programs Using Software Fault Trees,” IEEE Software, July 1991.

Magison, E.C., “Make Sure Your System is Safe,” Instrument & Control Systems (I&CS), December, 1979.

MIL-STD, See U.S. Department of Defense.

Musa, J., Iannino, A., and Okumoto K., *Engineering and Managing Software with Reliability Measures*, New York, NY: McGraw-Hill, 1987.

Ontario Hydro and Atomic Energy of Canada, Ltd., “Procedure for Systematic Design Verification of Safety Critical Software—Draft,” Ontario Hydro, Toronto, Ontario, Canada, February 28 ,1992.

Paques, Joseph-Jean, “Basic Safety Rules for Using Programmable Controllers,” Instrument Society of America (ISA) Transactions, Vol. 29, No. 2, 1990.

Paques, Joseph-Jean, “The Elements of Safety for Using Programmable Controllers,” Instrument Society of America (ISA) Transactions—Control Systems Safety, Vol. 30, No. 1, 1991.

Parnas, D. L., “Proposed Standard for Software for Computers in the Safety Systems of Nuclear Power Stations (based on IEC Standard 880),” TRIO, Computing and Information Science, Kingston, Ontario, March 26, 1991.

Patterson, David A., and Hennessy, John L., *Computer Architecture—A Quantitative Approach*, 1st ed., Morgan Kaufmann Publishers, San Mateo, CA, 1990.

Pilsworth, R., “Software Standards for Defense Procurement,” Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, NY, 1988.

Pofahl, Ekkehard, “The TUV Test Procedure for Programmable Logic Controllers,” Dipl.-Phys., TUV Rheinland, Cologne, West Germany, Rheinland Technical Inspection Agency.

Preckshot, G.G., “Data Communications Systems in Nuclear Power Plants,” UCRL-ID-114564, Lawrence Livermore National Laboratory, Livermore, California, May 1993a.

Preckshot, G.G., “Data Communications,” UCRL-ID-114567, Lawrence Livermore National Laboratory, Livermore, California, May 1993b.

Preckshot, G.G., “Real-Time Performance,” UCRL-ID-112616, Lawrence Livermore National Laboratory, Livermore, California, May 1993c.

Rondeau, Dan M., and Blackledge, Michael A., “The Preferred Process: Software Development,” Document Number SAND90-2227/7, Sandia National Laboratories, Albuquerque, New Mexico, 1991.

Sandia National Laboratories, Albuquerque, New Mexico.
“Sandia Software Guidelines Volume 3: Standards, Practices, and Conventions,” Document Number SAND85-2346, 1986.
“Sandia Software Guidelines Volume 1: Software Quality Planning,” Document Number SAND85-2344, 1987a.
“Sandia Software Guidelines Volume 5: Tools, Techniques, and Methodologies,” Document Number SAND85-2348, 1987b.

Saudi Aramco Material Specification, “Programmable Controller Based ESD Systems—Draft Issue,” Number 34-AMSS-623, September 1990.

Shooman, M. L., *Software Engineering: Design, Reliability, and Management*, New York, NY: McGraw-Hill, 1983.

Smith, Michael W., “PLC Emergency Shutdown Systems Used with a DCS in a Pilot Plant Environment,” Instrument Society of America (ISA), Paper #91-0599, 1991.

Smith, Steven E., “Fault Coverage in Plant Protection Systems,” Instrument Society of America (ISA) Transactions—Control Systems Safety, Vol. 30, No. 1, 1991.

Sparkman, D., “Techniques and Processes in Software Safety and Reliability: Part One,” Version 2.0, Lawrence Livermore National Laboratory, Nuclear System Safety Program, February 7, 1992.

Spiker, Rolf, “Are All Emergency Shut-Down Technologies Safe Enough?” Preliminary, Industrial Automation Inc., Houston, Texas, October 1990.

Triconex Corporation, “Tricon Fault-Tolerant Control: Technical Product Guide,” Irvine, CA, 1990.

U.S. Department of Defense
DOD 2167A, “Defense System Software Development,” ,February, 1988.
MIL-STD-483A, “Configuration Management Practices for Systems, Equipment, Munitions, and Computer Programs,” September 11, 1989.
MIL-STD-499A, “Engineering Management,” May 1, 1974.
MIL-STD-1456A, “Configuration Management Plan,” Department of the Air Force, June 4, 1985.
MIL-STD-1521B, “Technical Reviews and Audits for Systems, Equipment, and Computer Software,” Department of the Air Force, June 4, 1985.

Vora, Deepak D., “Engineering Contractor’s Perspective on Programmable Logic Controller,” Instrument Society of America (ISA), Paper #91-0531, 1991.

Walczak, Thomas A., “Emergency PLC Controlled Shutdown,” Instrument Society of America (ISA), Paper #90-609, 1990.

Wilhelm, R.E. Jr., *Programmable Controller Handbook*, New York, NY: Hayden, 1985.